Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования
“Новгородский государственный университет
имени Ярослава Мудрого”

на правах рукописи

МИХАЙЛОВ ДМИТРИЙ ВЛАДИМИРОВИЧ

ТЕОРЕТИЧЕСКИЕ ОСНОВЫ, МЕТОДЫ И АЛГОРИТМЫ ФОРМИРОВАНИЯ ЗНАНИЙ О СИНОНИМИИ ДЛЯ ЗАДАЧ АНАЛИЗА И СЖАТИЯ ТЕКСТОВОЙ ИНФОРМАЦИИ

Специальность 05.13.17 – Теоретические основы информатики

Диссертация на соискание ученой степени
доктора физико-математических наук

Научный консультант:
д.т.н., проф. ***Г.М.Емельянов***

Великий Новгород
2012

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ

АРМ	автоматизированное рабочее место
АФП	анализ формальных понятий
ГСС	глубинная синтаксическая структура
ЕЯ	естественный язык
ИГ	именная группа
ЛЗ	лексическое значение
ЛСК	лексическая синонимическая конструкция
ЛФ	лексическая функция
МУ	модель управления
НОПП	наибольшее общее подпонятие
НОСП	наименьшее общее суперпонятие
РЗ	расщепленное значение
РОСС	русский общесемантический словарь
РПЗ	расщепленное предикатное значение
СГ	семантический граф
СемП	семантическое представление
СК	семантический класс
СХ	семантическая характеристика
СЭ	семантическая эквивалентность
СЯУ	ситуация языкового употребления
ТКС	толково-комбинаторный словарь
ФП	формальное понятие
ХФ	характеристическая функция

ПЕРЕЧЕНЬ ОСНОВНЫХ ОБОЗНАЧЕНИЙ

$\Gamma = (W_R, V_R, \varphi, \Pi)$	Δ-грамматика, V_R – словарь пометок на ветвях дерева, W_R – словарь пометок на узлах, φ – матрица ограничений на размещение пометок на ветвях, Π – множество правил преобразований деревьев
$Rap(rule_j)$	условие применимости правила $rule_j \in \Pi$
C_0	ключевое слово комплекса единиц $wr_k \in W_R$ и связей $vr_j \in V_R$ между ними, заменяемых некоторым $rule_i \in \Pi$
$Lm(w_i)$	теория лексического значения слова w_i
$K = (G, M, I)$	формальный контекст с множеством объектов G и множеством признаков M, $I \subseteq G \times M$
$\leq$	отношение порядка для формальных понятий
$\Re$	решетка формальных понятий
$Null$	формальный контекст с пустыми множествами объектов и признаков
$\bullet$	операция конкатенации
$norm$	функция, ставит в соотвествие слову начальную форму
p_y	предлог между синтаксически главным и зависимым словом
$Spv : (v_{11}, v_{12}) \to v_{21}$	функция, ставит в соответствие расщеплённому предикатному значению $\{v_{11}, v_{12}\}$ его однословное выражение v_{21}
J	индексное множество
$Ls(Ts_i)$	модель линейной структуры ЕЯ-фразы Ts_i
$S = (O, R, Ts)$	ситуация языкового употребления, O – множество значимых в ситуации понятий, Ts – множество альтернативных форм описания ситуации в некоторой знаковой системе; $R \subset O^n$, $n \in 1, \ldots, \lvert O \rvert$

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Диссертация посвящена решению комплексной научно-технической проблемы унификации структуры и автоматизации пополнения предметных и языковых знаний для совокупности задач оценки семантической схожести и компрессии текстов предметно-ограниченного естественного языка без потери полезной смысловой составляющей. Предлагаются методы и алгоритмы формирования знаний о синонимии в виде классов решётки формальных понятий на основе ситуаций употребления предметно-ограниченного естественного языка для описания фрагментов действительности. В данной работе впервые предложено одновременное формирование предметных и языковых знаний непосредственно по текстам, вводимым пользователем без специальной подготовки в области языкознания.

Актуальность работы. Алгоритмически разрешимые процедуры распознавания смысла высказываний естественного языка (ЕЯ), а также способы представления смысла для решения практических задач составляют основу реализации интеллектуальных систем распознавания и синтеза речи, текста и изображений. Разработка таких систем относится к позиции “технологии обработки, хранения, передачи и защиты информации” перечня критических технологий федерального уровня от 21 мая 2006 года и образует самостоятельное направление, получившее название “Обработка естественного языка” [10, с. 27–48; 11, с. 79–84; 12, с. 165–209; 39, с. 81–219; 43, кн. 1, с. 9–139, 201–261; 108, с. 335–488; 116, с. 10, 44–55; 126, с. 27–28, 483–519; 133, с. 10–20].

Сферой рассмотрения автора настоящей диссертационной работы являются задачи, требующие установления полной или частичной эквивалентности по смыслу (семантической эквивалентности – СЭ) высказываний (текстов) ЕЯ [148, 151]. К числу таких задач можно отнести применение заданий открытой формы в системах компьютерного дистанционного обучения и контроля знаний [1, с. 55–69; 54, с. 16–18; 60, с. 117–120; 98; 102; 105; 129, с. 181–190], поиск изображений и распознавание семантики сложных информационных объектов по вербальному

описанию [117, 144, 147, 151], анализ сходства текстовых документов [38, 42]. Представление знаний в виде классов семантической эквивалентности текстов, которыми описываются фрагменты действительности, позволяет простым и естественным путём разделять вводимые в ЭВМ знания на уровни (языковой, предметный и т.п.) с учётом основной когнитивной (гносеологической) функции языка как средства передачи знаний от человека к человеку и инструмента для формирования новых знаний [21, с. 24–61]. При этом в качестве исходных данных для формирования знаний выступют тексты на предметно-ограниченном естественном языке, которые вводятся оператором без специальной подготовки в области прикладной и математической лингвистики.

Объект исследования настоящей диссертационной работы – программные средства распознавания, анализа и сжатия текста на естественном языке.

Предметом исследования являются методы и алгоритмы формирования знаний о синонимии.

Областью непосредственного применения теоретических результатов работы является автоматизированный контроль знаний. Важное преимущество автоматизированного обучения состоит в реализации известного педагогического принципа индивидуализации обучения [1, с. 227]. При этом наибольший интерес представляют задания открытой формы, то есть задания, требующие самостоятельного формулирования ответа на вопрос теста. В отличие от заданий закрытой формы (выбор правильного ответа из набора вариантов), заданий на соответствие, заданий на установление правильной последовательности, тесты открытой формы исключают догадку [60, с. 160] и позволяют максимально приблизить компьютерный тест к традиционному взаимодействию “Учитель–Ученик”.

Однако имеются недостатки, в силу которых тестовые задания открытой формы не нашли широкого применения в системах контроля знаний. Эффективная реализация открытых тестов, как показано в [98], предполагает известную структуру ЕЯ-форм выражения знаний эксперта. Сами открыте тесты зачастую сводятся либо к простым заданиям на дополнение с ограничениями на ответы [1,

с. 55–56; 54, с. 18; 60, с. 117], либо к простому поиску среди “правильных” вариантов [102]. Причина кроется в нетехнологичности заданий открытой формы. Допуская свободное формулирование ответа, испытуемые могут использовать синонимы, а также изменять порядок следования слов, что особенно актуально для естественных языков со свободным порядком слов в предложении. Основными требованиями здесь являются способность системы анализировать СЭ высказываний с отклонениями от грамматической нормы, единообразие механизмов оперирования предметными и языковыми знаниями, а также ориентацию на автоматизированное пополение последних с минимумом трудозатрат.

Следует отметить, что к настоящему моменту серьёзных попыток смоделировать на ЭВМ формирование знаний о синонимии в ЕЯ во взаимосвязи с процессом накопления знаний о языке в целом и об окружающем мире не предпринималось, несмотря на многочисленные публикации, посвящённые:

– синтаксису, его связи с семантикой и лексическими средствами языка, реализующими механизм синонимического перифразирования. Как наиболее близкие рассматриваемой в диссертации преоблеме здесь следует отметить работы Мельчука И.А. [62, 162], Гладкого А.В. [14, 15], Апресяна Ю.Д. [3], Кибрика А.Е. [45, 97], Тестельца Я.Г. [121], Солганика Г.Я. [118], Тузова В.А. [123];

– компьютерным словарям, тезаурусу и машинному фонду русского языка. Наибольший интерес в этом направлении заслуживают работы Караулова Ю.Н. [44], Нариньяни А.С. [100], Рубашкина В.Ш. [111], Попова Э.В. [106], Леонтьевой Н.Н. [58], Демьянкова В.З. [21,22], Гусева В.Д. [18];

– информационному поиску, где следует отметить работы Леонтьевой Н.Н. [56,58], Осипова Г.С. [101], Попова Э.В. [106], Фомичёва В.А. [124,125,152], Соснина П.И. [119,182], Тихомирова И.А. [122], Журавлёва Ю.И. [38], Игнатова Д.И. [42], Гуревича И.Б. [155], Мучника И.Б. [5], Райгородского А.М. [16] и ряд других [17,134–138,142,160,161,180,181,187].

Г.М. Емельяновым, Т.В. Кречетовой и Е.П. Курашовой была предпринята попытка решить эту задачу с привлечением уровня глубинного синтаксиса ЕЯ на

основе модели СЭ с использованием грамматик деревьев (Δ-грамматик) в качестве аппарата математического моделирования [151]. Указанный математический аппарат, предложенный А.В. Гладким и И.А.Мельчуком в [14,15] и расширенный разделением преобразований узлов и ветвей, позволил решить задачу моделирования синонимических преобразований ЕЯ-высказываний на уровне варьирования универсальной (абстрактной) лексикой без существенного ограничения входного ЕЯ и предметной области решаемых задач. Но и данному подходу в том виде, в котором он описывается в [151], присущи серьёзные недостатки:

– на уровне глубинного синтаксиса текст представлен фразами, каждая из них соответствует простому распространённому предложению. При этом нельзя говорить о необходимых и достаточных признаках синонимии по анализу применимости правил и целесообразности трансформаций того или иного типа;

– словарная подсистема предполагается замкнутой ввиду существенной сложности описываемой словарём информации;

– отсутствует формализация компонент условий применимости правил синонимических преобразований глубинных синтаксических структур;

– синонимические преобразования деревьев глубинного синтаксиса в теоретическом плане проработаны не до конца. Использованный в [151] набор правил был взят из работ Ю.Д. Апресяна [3] и И.А. Мельчука [62]. По оценке последнего, указанные правила не претендуют на полноту и возможно их расширение по результатам соответствующих исследований.

Современные поисковые системы, анализируя ЕЯ-запрос, используют статистику встречаемости слов запроса в различных контекстах с учётом возможных синонимов с целью поиска документа, максимально релевантного запросу [5,17]. Аналогичный принцип используется и в статистическом переводе, в частности, в составе поисковой системы Яндекс [134]. Данный подход полностью оправдывает себя в задаче информационного поиска, но он не позволяет воссоздать целостный образ самой ситуации использования ЕЯ для описания фрагмента действительности. Сказанное особенно актуально, в частности, при подготов-

ке тестовых заданий открытой формы, когда задача является принципиально обратной: известен фрагмент реальности и разработчику теста требуется выделить все возможные формы описания этого фрагмента в заданном ЕЯ. При этом также крайне необходима двусторонняя связь “носитель ЕЯ (разработчик теста) – база знаний” с поддержкой актуального (в терминологии баз данных, см. [20, с. 46]) состояния целостного образа отражения фрагмента действительности в сознании разработчика и в его языке, что позволило бы вести сравнительный анализ уровня владения заданным естественным языком и предметными знаниями у разработчиков тестов по некоторой заданной предметной области.

Таким образом, задача разработки эффективных средств машинного представления знаний о СЭ в совокупности с реализацией механизма взаимодействия языковых и предметных знаний является чрезвычайно *актуальной*.

С учетом обозначенной выше проблемы СЭ и её значимости для компьютерной лингвистики в целом, **цель диссертационной работы** сформулирована как *разработка и теоретическое обоснование структуры знаний о синонимии, а также методов и алгоритмов их формирования и использования для совокупности задач оценки семантической схожести текстов предметно-ограниченного естественного языка, автоматизации пополнения и компрессии баз языковых и предметных знаний*.

Для достижения указанной цели в диссертации ставятся и решаются следующие **задачи**:

1) Анализ существующих методов формализации семантики конструкций ЕЯ и определение общих требований, предъявляемых к механизму сравнения смыслов на функциональном уровне.

2) Разработка и исследование методов анализа СЭ на уровне варьирования абстрактной лексикой.

3) Разработка методов автоматизированного формирования и кластеризации знаний о семантике конструкций предметно-ограниченного естественного языка с учётом взаимосвязи языковых уровней.

4) Исследование и алгоритмизация механизма использования морфологии и синтаксиса ЕЯ для задач кластеризации, разделения и сжатия баз предметных и языковых знаний.

5) Разработка и исследование методов численной оценки семантической схожести текстов предметно-ограниченного естественного языка.

6) Разработка архитектуры программной системы, реализующей предложенные принципы, методы и алгоритмы.

Методы исследования. Для решения поставленных в работе задач были использованы методы формальной теории языков, математической логики и теории множеств, теории решеток и анализа формальных понятий, системной типологии языков и когнитологии, основные положения теоретической и когнитивной лингвистики, а также прикладные методы анализа данных и знаний.

Научная новизна. В диссертации разработаны теоретические основы автоматизированного формирования знаний о синонимии и их использования для сокращения объёмов баз предметных и языковых знаний в задачах анализа текстов. В частности, новыми являются следующие результаты:

- методика автоматизированного формирования и экспериментальной оценки знаний выделением классов семантической эквивалентности текстов, учитывающая целостный образ ситуации употребления предметно-ограниченного подмножества ЕЯ для описания факта действительности;
- подход к решению задачи распознавания сверхфразовых единств в текстах на уровне глубинного синтаксиса. При этом динамическая информационная модель совокупности правил грамматики деревьев сводит поиск последовательности преобразований с заданными свойствами к известным задачам теории сетей Петри;
- принцип выделения и кластеризации семантических отношений как теоретическая основа формирования смыслового эталона на множестве эквивалентных по смыслу фраз предметно-ограниченного подмножества естественного языка;

- метод и алгоритмы автоматизированного формирования смыслового эталона на множестве СЭ-фраз в виде решётки формальных понятий, а также метод компрессии текстовой базы знаний на основе выделенных эталонов;
- метод численной оценки семантической схожести текстов предметно-ограниченного ЕЯ с учётом разделения языковых и предметных знаний;
- типовая архитектура программной системы контроля знаний, реализующая предложенные в работе принципы, методы и алгоритмы.

Достоверность теоретических результатов обеспечивается применением апробированного математического аппарата, корректностью изложения основных теоретических положений работы с формулировкой необходимых утверждений, лемм и теорем, строгостью математических доказательств, согласованностью с ранее полученными результатами других авторов. Теоретические положения иллюстрируются примерами реализации компонент программной системы тестирования знаний и решения возникающих при этом инженерных задач.

Апробация работы. Полученные результаты апробированы в докладах на конференциях, семинарах и конгрессах:

- 5-й, 6-й, 7-й, 8-й, 9-й Международных конференциях “Распознавание”, Курск, 2001, 2003, 2005, 2008, 2010;
- 10-й, 12-й, 13-й, 14-й, 15-й Всероссийских конференциях “Математические методы распознавания образов”, Москва, 2001, 2005, Зеленогорск (Ленинградская область), 2007, Суздаль (Владимирская область), 2009, Петрозаводск, 2011;
- VI-й, VIII-й Всероссийских конференциях “Методы и средства обработки сложной графической информации”, Нижний Новгород, 2001, 2005;
- Международном семинаре Диалог’2002 “Компьютерная лингвистика и интеллектуальные технологии”, Москва, 2002;
- 4-й, 5-й, 6-й, 7-й, 8-й Международных конференциях “Интеллектуализация обработки информации”, Алушта (Автономная Республика Крым, Украина), 2002, 2004, 2006, 2008, Пафос (Республика Кипр), 2010;

- 6-й, 7-й, 8-й, 9-й, 10-й Международных конференциях “Распознавание образов и анализ изображений: новые информационные технологии”, Великий Новгород, 2002, Санкт-Петербург, 2004, 2010, Йошкар-Ола, 2007, Нижний Новгород, 2008;
- VI-м Международном конгрессе по математическому моделированию, Нижний Новгород, 2004;
- XVIII Международной научно-методической конференции “Математика в вузе”, Санкт-Петербург, 2005;
- 6-й, 7-й, 8-й, 9-й Международных научно-технических конференциях “Интерактивные системы: проблемы человеко-компьютерного взаимодействия”, Ульяновск, 2005, 2007, 2009, 2011;
- XIII-й, XIV-й, XV-й, XVI-й, XVII-й, XVIII-й научных конференциях преподавателей, аспирантов и студентов НовГУ “Дни науки в НовГУ”, Великий Новгород, 2006, 2007, 2008, 2009, 2010;
- юбилейной научно-практической конференции “Великий Новгород – город университетский”, Великий Новгород, 2003;
- научных семинарах кафедр “Программного обеспечения вычислительной техники и автоматизированных систем” и “Информационных технологий и систем” Новгородского государственного университета имени Ярослава Мудрого с 2001 по 2012 годы.

Публикации. Материал настоящей работы основан на публикациях [8, 29–35, 41, 47–52, 63–96, 113, 115, 130, 131, 143, 145, 146, 149, 150, 163–178]. Всего по теме диссертации опубликовано 75 работ, в том числе монография [87], 18 статей в журналах из перечня ВАК [48, 71, 79, 81, 84, 85, 96, 143, 146, 150, 164, 165, 167, 170, 172, 174, 176, 178]. Имеется свидетельство о регистрации программы для ЭВМ [113]. В трудах международных конференций представлено 28 работ, а именно: [29–32, 35, 47, 49, 52, 63, 66, 69, 70, 75, 82, 86, 89, 92, 94, 145, 149, 163, 166, 168, 169, 171, 173, 175, 177], работы [64, 67, 68, 73, 76, 77, 83] опубликованы в сборниках трудов всероссийских конференций.

Структура и объём диссертации. Диссертационная работа включает в себя: введение, шесть глав, заключение, список литературы и два приложения. Общий объем диссертации составляет 333 страницы машинописного текста. Основная часть работы изложена на 237 страницах и содержит 78 рисунков и 15 таблиц. Список литературы включает 188 наименований.

На защиту выносятся следующие основные положения:

1. Методика формирования и экспериментальной оценки знаний, основанная на концепции ситуации употребления естественного языка как единицы формализованного описания его семантики.
2. Подход к нахождению системы целевых выводов в Δ-грамматике как основа выделения сверхфразовых единств в текстах на уровне глубинного синтаксиса.
3. Принцип формирования и кластеризации семантических отношений как основы классов СЭ.
4. Метод и алгоритмы выделения смыслового эталона на множестве эквивалентных по смыслу фраз предметно-ограниченного естественного языка.
5. Численная оценка семантической схожести текстов предметно-ограниченного ЕЯ относительно ситуаций его употребления.
6. Метод компрессии текстовой базы знаний с применением смысловых эталонов.

Указанные теоретические положения составляют суть решения научной проблемы автоматизации накопления информации о языке как средстве передаче знаний от человека к человеку, имеющей важное значение для обработки данных на ЭВМ в социально-экономических, научных и культурных задачах.

Краткое содержание работы по главам.

В первой главе формулируются требования к процессу накопления знаний о синонимии как основы кластеризации предметных и языковых знаний. Вводится понятие ситуации языкового употребления в качестве единицы формализованного описания семантики ЕЯ. В рамках теории анализа формальных понятий сформу-

лирован и обоснован принцип формирования и экспериментальной оценки знаний в виде классов смысловой эквивалентности текстов на основе ситуаций употребления предметно-ограниченного подмножества ЕЯ. При этом особую роль в представлении знаний о синонимии играет уровень глубинного синтаксиса, для которого рассматривается система синонимических преобразований над деревьями глубинных синтаксических структур. Далее в главе рассматриваются достоинства и недостатки установления СЭ на основе указанных синонимических преобразований, формулируются задачи, решаемые в последующих главах.

На основе полученного в первой главе теоретико-множественного описания процесса установления СЭ **во второй главе** исследуется проблема полноты представления смысла при формировании прецедентов ситуаций синонимии для уровня абстрактной лексики. При этом содержательную основу сжатия смысловой информации составляют сверхфразовые единства на уровне глубинного синтаксиса. Ставится и решается задача их выделения и формального представления для ЕЯ-высказываний, состоящих более чем из одного простого распространенного предложения. Рассмотрены вопросы эффективного использования знаний уровня глубинного синтаксиса при решении указанной задачи.

Третья глава посвящена вопросам автоматизации формирования и классификации семантических отношений в текстах как основы знаний о синонимии. С целью формализации условий применимости правил синонимических преобразований деревьев глубинного синтаксиса в рамках подхода “Смысл⇔Текст”, рассмотренного в первых двух главах, предложено описание толкования лексического значения слова на языке логики предикатов 1-го порядка. Исследованы принципы обобщения независимых вариантов толкований значения слова относительно заданного предметно-ориентированного подмножества ЕЯ. Для автоматизации получения толкований значений слов предложена комплексная методика выделения и классификации отношений, необходимых для ролевой идентификации сущностей относительно заданной ситуации, а также для построения тезауруса заданной предметной области. Предложен принцип формирования указан-

ных отношений на основе множеств СЭ-фраз, в составе каждого из которых ЕЯ-фразы описывают одну и ту же ситуацию действительности. В **приложении 1** представлены фрагменты исходного текста программы на языке Visual Prolog 5.2, реализующей указанный принцип.

Четвертая глава диссертации посвящена вопросам формирования знаний в рамках ситуаций употребления ЕЯ по результатам синтаксического разбора текстов внешней программой синтаксического анализа. Рассматривается использование синтаксического контекста существительного в качестве основы семантической кластеризации текстов. На основе свойств соотношения смыслов соподчиненных слов решается задача накопления знаний о частичных смысловых эквивалентностях ЕЯ-высказываний.

В пятой главе синтаксический контекст существительного анализируется в рамках принципов формирования семантических отношений, разработанного в третьей главе. Предложено представление ситуации языкового употребления формальным контекстом в качестве информационной единицы тезауруса предметной области. Описана методика построения указанного формального контекста по результатам синтаксического разбора семантически эквивалентных ЕЯ-фраз. Далее в главе представлен предложенный автором метод оценки схожести формальных контекстов ситуаций языкового употребления и описываются правила установления семантической эквивалентности фраз относительно заданного предметно-ориентированного подмножества естественного языка.

Шестая глава диссертации посвящена решению задачи разделения и сжатия баз предметных и языковых знаний с применением комплексной методики формирования и кластеризации семантических отношений, изложенной в **разделах 3.5**, **4.1**, **5.2** и **5.3**. Вводится понятие смыслового эталона для СЯУ и рассматриваются два приближённых метода построения такого эталона. Описывается методика интерпретации ответа испытуемого на тестовое задание открытой формы, а также подход к минимизации текстовой базы знаний на основе выделения эталонов ситуаций языкового употребления. Завершает главу описание архи-

тектуры системы контроля знаний, реализующей предложенные в диссертации принципы, методы и алгоритмы. Документы, подтверждающие апробацию системы, представлены в **приложении 2**.

В **заключении** работы сформулированы основные научные и практические результаты диссертационной работы.

В приложении 1 приведены фрагменты текста программы на языке Visual Prolog 5.2, осуществляющей генерацию теоретико-решёточного представления ситуации языкового употребления на основе множества семантически эквивалентных ЕЯ-фраз с построением объектно-признаковых описаний смысловых отношений в рамках эталона СЯУ. Представлена реализация процедур выделения основ посимвольным сравнением слов различных фраз, таксономии буквенных инвариантов в составе отдельных слов при выявлении основ (с учетом возможных синонимов и частичных совпадений буквенного состава основ у слов с разным лексическим значением), а также методов оценки качества такой таксономии.

Теоретическая и практическая значимость. Диссертационная работа носит теоретико-прикладной характер. Полученные в ней результаты, разработанные методы и реализующие их программы могут быть использованы для решения широкого класса задач обработки текстов, а также сжатия информации без потери полезной смысловой составляющей. Наряду с ЕЯ-текстами, выделение смысловых эталонов предлагаемыми в работе методами актуально для задач распознавания и анализа семантики любых сложных информационных объектов, в частности, изображений, при формировании баз данных и знаний. Предложенная в третьей главе комплексная методика выделения и кластеризации семантических отношений и её программная реализация, представленная в **приложении 1** фрагментами исходного текста на языке Visual Prolog, могут быть перенесены на СЭ-представления информации произвольного рода о любых объектах либо ситуациях действительности с целью автоматического синтеза наиболее оптимального описания исследуемых объектов (ситуаций) на заданном формальном языке. В частности, результаты диссертационной работы реализованы в рамках следующих НИР:

1. Грант РФФИ № 03-01-00055-а “Разработка математического аппарата для распознавания сверхфразовых единств в текстах”, рук. Емельянов Г. М., отв. исп. Михайлов Д.В.
2. Грант РФФИ № 06-01-00028-а “Разработка методов автоматизированного пополнения тезауруса для задач распознавания смысловой эквивалентности текстов”, рук. Емельянов Г. М., отв. исп. Михайлов Д.В.
3. Грант РФФИ № 10-01-00146-а “Разработка методов автоматизированного накопления и систематизации знаний о морфологии и синтаксисе естественного языка для задач семантической кластеризации текстов”, рук. Емельянов Г. М., отв. исп. Михайлов Д.В., гос. рег. № 0120.1 164263, 2010-2012 г.
4. Грант № ТОО-3.3-408 Минобразования РФ, отв. исп. Михайлов Д.В.
5. Контракт № И 0675 ФЦП “Интеграция”, отв. исп. Михайлов Д.В., гос. рег. № 0120.0 300918.
6. ГБ НИР “Разработка и исследование математических моделей многопараметрических систем”, рук. Емельянов Г.М., отв. исп. Михайлов Д.В., по заданию Минобрнауки РФ, гос. рег. № 0120.0 704719, 2007-2011 г.

Область исследования согласно паспорту специальности 05.13.17 – “Теоретические основы информатики”:

- разработка и исследования методов и алгоритмов анализа текста (п. 5);
- разработка принципов и методов извлечения данных из текстов на естественном языке (п. 6);
- разработка основ математической теории языков и грамматик (п. 10).

Согласно формуле специальности “Теоретические основы информатики”, к ней относятся, в числе прочего, “исследования процессов создания, накопления и обработки информации; исследования методов преобразования информации в данные и знания; создание и исследование информационных моделей …”.

Диссертация посвящена исследованию процессов накопления знаний о синонимии в естественном языке, созданию и исследованию информационной модели указанного явления, разработке принципов и методов извлечения знаний, а

также программных средств автоматизации построения концептуальной модели предметной области на основе классов семантической эквивалентности для текстов предметно-ограниченного ЕЯ, что полностью соответствует паспорту специальности 05.13.17 – “Теоретические основы информатики”.

Личный вклад автора. В диссертационной работе обобщены результаты, полученные лично автором. Постановка и решение задачи распознавания сверхфразовых единств в текстах на уровне глубинного синтаксиса принадлежит автору. Решение задач формирования и кластеризации знаний на основе синтаксического контекста существительного предложено автором как обобщение результатов, полученных совместно с Н.А. Степановой. Теоретические основы формирования знаний о языке на основе ситуаций его употребления развиты автором совместно с А.Н. Корнышовым. Метод оценки семантичекой схожести текстов предметно-ограниченного ЕЯ, а также метод и алгоритмы выделения смыслового эталона на множестве эквивалентных по смыслу ЕЯ-фраз, метод компрессии текстовой базы знаний и подход к интерпретации ответа испытуемого на тестовое задание открытой формы (включая архитектуру программной системы контроля знаний) разработаны лично автором. Эксперименты на ЭВМ подготовлены и выполнены автором в рамках выпускных квалификационных работ студентов специальностей “Прикладная математика и информатика” и “Программное обеспечение вычислительной техники и автоматизированных систем”.

Глава 1
СИТУАЦИЯ ЯЗЫКОВОГО УПОТРЕБЛЕНИЯ И КЛАСТЕРИЗАЦИЯ ПРЕДМЕТНО-ЯЗЫКОВЫХ ЗНАНИЙ

Настоящая глава посвящена общей постановке задачи автоматизированного накопления знаний о синонимии как основы кластеризации предметных и языковых знаний. Формулируются общие требования к процессу установления семантической эквивалентности текстов относительно предметно-ограниченного подмножества естественного языка. Вводится понятие ситуации языкового употребления (СЯУ) как единицы формализованного описания семантики естественного языка. Строится теоретико-множественное описание процесса установления семантической эквивалентности с учетом выявленных функциональных требований. В рамках теории анализа формальных понятий сформулирован и обоснован принцип формирования и экспериментальной оценки знаний в виде классов смысловой эквивалентности текстов на основе ситуаций употребления предметно-ограниченного естественного языка для описания фактов действительности. Основные результаты главы опубликованы в [32,33,47,48,76,85,87].

1.1. Семантическая эквивалентность и ситуация языкового употребления

В настоящий момент в теоретической лингвистике и смежных с ней дисциплинах не существует общепризнанного и бесспорного определения языка как такового.

В частности, существует довольно распространенное понимание языка как сложной знаковой системы [3, 45, 128]. Различные знаковые системы являются предметом изучения семиотики [46, 55]. При этом сам естественный язык рассматривается с двух точек зрения [121].

С функциональной точки зрения строение ЕЯ определяется использованием последнего в качестве средства общения. Формальная точка зрения предполагает наличие у языка некоторой абстрактной модели, которая не зависит от конкретного способа использования ЕЯ и может быть описана формальной грамматикой. Моделирование естественных языков с помощью формальных грамматик, порождающих возможные высказывания, было предложено Н. Хомским [127]. Хорошим примером рассмотрения языка с функциональной точки зрения может послужить модель языка как преобразователя “Смысл⇔Текст” [62].

Совмещая точки зрения и подходы к описанию языка, естественный язык следует определить как сложную знаковую систему, основной функцией которой является использование в качестве средства общения между людьми. При этом абстрактная модель языка задается формальным механизмом порождения всех возможных высказываний в этой знаковой системе, а также механизмом установления соответствия высказываниям их смыслов плюс установление соответствия между самими смыслами. Под естественностью языка будем понимать наличие таких свойств, как синонимия слов и словосочетаний, а также свободный порядок слов в предложении [97, 99, 109].

Опираясь на данное определение ЕЯ, введем некоторые базовые термины для формального описания рассматриваемого нами процесса установления СЭ.

Определение 1.1. Под *конструкцией ЕЯ* (далее в работе мы будем также использовать термин *“языковая конструкция”*) в настоящей работе понимается последовательность знаков в некоторой знаковой системе, которая может быть использована для фиксации некоторого количества высказываний этого ЕЯ в памяти ЭВМ.

Определение 1.2. *Семантическими знаниями* мы будем называть языковые знания, необходимые для использования некоторого ЕЯ в процессе общения. Соответственно, *носителем языка* следует считать обладателя семантических знаний.

Следствие. Под *семантическим отношением* следует понимать некоторую универсальную связь, усматриваемую носителем языка в тексте. Именно таким образом понимается семантическое отношение в идеологии Русского общесемантического словаря (РОСС) [58].

Смысл высказывания представляет собой довольно сложный и удаленный от уровня наблюдения конструкт [103].

Строгое формальное определение смысла, которое автором настоящей работы использовано в рамках предлагаемых идей, методов и алгоритмов, будет дано в главе 3. Здесь мы остановимся на следующем определении в первом приближении, приемлемом с точки зрения практики обработки текста.

Определение 1.3. Под *смыслом* ЕЯ-высказывания понимается информация, содержащаяся в высказывании и не меняющаяся при его синонимических преобразованиях [62, с. 10–11]. Иными словами, смысл – это информация о том, как объект или ситуация реального мира отражается в сознании говорящего.

Рассматривая текст как поверхностную форму фиксации высказываний на ЕЯ и единственный способ выражения смысла в процессе общения с ЭВМ на этом ЕЯ, то есть допуская знаковую систему в качестве единственного средства выражения смысла, будем считать, что понятие смысловой эквивалентности совпадает с понятием семантической эквивалентности.

При этом задача установления семантической эквивалентности ЕЯ-высказываний состоит в сравнении информации, отвечающей *определению 1.3*, посредством обработки конструкций ЕЯ, которые эту информацию фиксируют [111]. Семантическую эквивалентность, таким образом, в общем случае следует понимать как теоретико-множественное пересечение смыслов.

Исходя из сформулированного нами определения ЕЯ как сложной знаковой системы, в качестве единицы формализованного описания семантики конструкций ЕЯ для решения задачи установления СЭ будем использовать модель ситуаций употребления ЕЯ (ситуаций языкового употребления). Данная модель предложена А. Н. Корнышовым в [48] совместно с автором настоящей диссерта-

ционной работы как основа концептуально-ситуационного моделирования ЕЯ-высказываний.

Предназначение СЯУ состоит в разделении языкового опыта в соответствии с разделением концептуальной картины мира. Подобное разделение лежит в основе генезиса ЕЯ [61]. Ситуации языкового употребления рождаются из потребности обозначить и описать новый социальный опыт либо содержание обстоятельств типичных совместных действий [107] посредством ЕЯ.

Определение 1.4. Под ситуацией языкового употребления (ситуацией употребления ЕЯ) понимается описание нового социального опыта (содержания совместных действий) средствами этого ЕЯ. Данное описание выполняется в некоторой знаковой системе с целью обобщения и передачи знаний от человека к человеку.

Формально фиксируемый ситуацией S языковой контекст представляется тройкой:

$$S = (O, R, Ts), \tag{1.1}$$

где O есть множество символов, отождествляемых с некоторыми понятиями; Ts – множество альтернативных форм описания ситуации в некоторой знаковой системе; $R \subset O^n$, где $n \in 1,\ldots,|O|$.

Следует отметить, что посредством модели (1.1) могут быть представлены любые семантические знания о заданном ЕЯ.

Действительно, конкретный вид элементов множества Ts не определен, что позволяет представлять формы языкового описания S, в частности, деревьями синтаксического подчинения. А поскольку синтаксические отношения задают синтагматические зависимости, которые определяют возможность сосуществования словоформ в линейном ряду, то допускается приводить элементы множества Ts к естественному для поверхностного уровня ЕЯ представлению в линейной форме. В качестве элементов Ts в работе рассматриваются совокупности символьных цепочек (содержательно – ЕЯ-фразы), причём для $\forall Ts_i \in Ts \quad \exists Tr_i$:

$Ts_i = Synt(Tr_i)$, где Tr_i есть ориентированное помеченное дерево, а $Synt$ – сюръективная функция, определяемая правилами синтаксиса языка.

Отношения из множества R, как и формы описания ситуации, представляемые множеством Ts, также могут быть любого типа, что позволяет описывать посредством тройки (1.1) любые преобразования конструкций заданного ЕЯ. Согласно *следствию определения 1.2*, синтаксические зависимости можно рассматривать как частный случай семантических отношений, что дает возможность решать задачу формирования и классификации произвольных отношений относительно различных ситуаций вида (1.1). Этот вопрос более подробно освещается в пятой главе работы.

Модель (1.1) учитывает как синтагматические отношения между языковыми единицами, которые задаются с помощью множества *R*, так и парадигматические отношения, которые задаются варьированием элементов множества *O*. Кроме того, смысл ситуации S отделен от множества форм поверхностного выражения данной ситуации. Благодаря такому разделению допускается сравнение смыслов без порождения всех возможных инвариантных по смыслу фраз.

1.2. Концептуальная модель процесса установления семантической эквивалентности

Опираясь на введенное в предыдущем разделе представление о СЯУ как основе формализованного описания семантики ЕЯ в задаче установления СЭ, настоящий раздел имеет целью описание данной задачи на функциональном уровне и установление границ проблемной области сравнения смыслов.

Рассмотрим компонент O тройки (1.1) с точки зрения формирования множества R на основе Ts. В общем случае $O = M \cup V$, где для $\forall o_j \in M$ найдётся $o_k \in V$ такое, что понятию o_j соответствует дочерний узел с пометкой w_j, а понятию o_k – родительский узел с пометкой w_k в некотором Tr_i: $Ts_i = Synt(Tr_i)$,

$Ts_i \in Ts$, а $M \cap V \neq \varnothing$. В таких случаях далее будем говорить, что слово, соответствующее символьной цепочке w_j, подчинено (синтаксически) слову, соответствующему цепочке w_k. При этом задача СЭ формулируется следующим образом.

Задача 1.1. Дано множество ЕЯ-текстов G. Требуется: по результатам разбора каждого $g_i \in G$ выявить множества $V(g_i)$ и $M(g_i)$, а также тернарное отношение $I \subseteq G \times M \times V$: $M = \bigcup_i M(g_i)$, $V = \bigcup_i V(g_i)$. Далее на основе I необходимо сформировать множество R и выделить группы текстов по сходству встречаемости понятий в одних и тех же $r_j \in R$. В конечном итоге требуется доказать идентичность ролей сходных понятий относительно сходных ситуаций, описываемых сравниваемыми текстами.

Наиболее близка данной идее обработка текстов на основе коммуникативной грамматики. Хорошим примером является поисковая система Exactus [122].

Тем не менее существуют задачи сравнения смысла, отличные от традиционного для поисковых систем взаимодействия “запрос – ответ”.

Примером является интерпретация текста ответа на тестовое задание открытой формы в системе автоматизированного контроля знаний. Необходимо не столько отобразить ответ на предметную область, сколько оценить его близость ответу, “правильному” с точки зрения преподавателя, конструировавшего тест. Анализ близости высказываний здесь требует учета лексико-функциональной синонимии, в частности – расщепленных значений и конверсивов [62]. В более общем случае многих обучаемых мы имеем задачу текстовой кластеризации [167].

По оценке Г. С. Осипова [101, с. 33], требуется более детальное исследование свойств связей между минимальными семантико-синтаксическими языковыми единицами в рамках самой коммуникативной грамматики.

Как следует из определения, сформулированного нами в предыдущем разделе главы, задача установления СЭ условно разбивается на две подзадачи: задачу восприятия текстов и задачу сравнения семантических представлений входных текстов. Согласно *определению 1.2*, процесс решения задачи восприятия тек-

ста строится на семантических знаниях. Задача сравнения семантических представлений решается посредством той части абстрактной модели языка, которая обеспечивает соответствие между смыслами (переход от одного семантического представления к другому).

К основным проблемам при выбранном теоретико-философском подходе можно отнести следующее.

Во-первых, будучи концептуальной моделью в первом приближении, модель вида (1.1) должна быть по-настоящему формализована. Это означает, в частности, формализацию представления каждой из её компонент для различных языковых уровней, поскольку природа отношений в составе множества R ничем не ограничена. Кроме того, необходима формализация взаимосвязей между самими языковыми уровнями, что моделью (1.1) не учитывается.

Во-вторых, необходимо разработать способы представления самих семантических знаний в системе и механизмы их пополнения. Семантические знания являются той базой, которая обеспечивает решение как задачи восприятия текстов, так и задачи сравнения их семантических представлений.

Основные достоинства выбранного подхода можно сформулировать следующим образом. Организация систем обработки ЕЯ-текстов на базе семантических знаний позволяет расширить возможности этих систем от жесткой ориентации на работу в предельно ограниченной предметной области. Это объясняется тем, что центральное место в семантических исследованиях большинства лингвистических теорий занимает не конкретная предметная лексика, а абстрактные слова (названия отношений, слова-кванторы), за счет которых обеспечивается богатое варьирование форм языкового описания для ситуации вида (1.1). Именно абстрактные слова должны в первую очередь подвергаться семантическому анализу [3, 44, 62].

Рассмотрим теперь, какие из выделенных нами требований к функционированию системы установления СЭ являются ключевыми для оценки адекватности рассматриваемых далее в работе принципов, методов и алгоритмов. Будем

вести рассуждения в предположении, что семантическая эквивалентность как явление описывается некоторой формальной моделью [148,151].

В общих чертах следует считать, что относительно заданного предметно-ориентированного ЕЯ-подмножества модель решает задачу установления СЭ, если она устанавливает семантическое тождество внешне различных предложений (синонимию) и анализирует грамматическую правильность предложений. В более общем случае отсутствия предметных ограничений модель должна также устанавливать семантическое различие внешне совпадающих предложений (омонимию).

Для формального описания отношений синонимии и омонимии между предложениями ЕЯ и распознавания грамматической правильности предложений необходим формальный аппарат лингвистических описаний [151]. Если естественный язык представить в виде формальной системы, то, согласно принятой нами идее семантики конструкции ЕЯ, он становится языком описания смыслов в формальной модели семантической эквивалентности. Подробнее об описании смыслов языковых конструкций на самом естественном языке мы остановимся в главе 3. Сейчас же мы сформулируем основные требования к языку формального описания и исчисления смыслов для задачи СЭ.

Во-первых, каждый комбинаторный тип цепочки в таком языке должен иметь один и только один смысл. При наложении ограничений предметного характера фразы ЕЯ при единственности синтаксической интерпретации могут обладать множественностью семантических интерпретаций, соответствующих смысловым оттенкам, но не нести взаимно исключающие смыслы. В этом случае понимание обеспечивается пресуппозицией [97].

Во-вторых, язык описания и исчисления смыслов должен быть языком универсальной канонизации, то есть накладываемые на язык ограничения не зависят от предметной области, которую этот язык описывает.

При этом сама модель СЭ должна быть такова, что любой ее компонент не только реализуем на ЭВМ, но и способен к расширению на основе входных текстов. Иными словами, модель должна быть обучаемой.

1.3. Уровень глубинного синтаксиса

В наибольшей степени требованиям, отмеченным в предыдущем разделе, отвечает модель языка как преобразователя “Смысл⇔Текст” [62]. Действительно, сам естественный язык в данном теоретическом подходе рассматривается как преобразователь текстов в смыслы и обратно. При этом смысл рассматривается как инвариант синонимических преобразований одних конструкций ЕЯ в другие, что позволяет выстраивать иерархию синонимических преобразований, решая задачу установления соответствия между смыслами. Предполагается, что сама синонимия языковых конструкций возникает не только за счет лексических синонимов, но и за счет синтаксических и лексически обусловленных вариантов высказывания.

В модели “Смысл⇔Текст” эти средства представлены в виде синтаксических и лексических правил перифразирования, базирующихся на аппарате лексических функций (ЛФ) [62].

Как отмечали И.А. Мельчук и А.К. Жолковский в [162, с. 77], “каждая ЛФ есть функция в математическом смысле, представляющая некоторый весьма общий смысл типа ‘очень’, ‘начинаться’ или ‘выполнять’, или же определенную семантико-синтаксическую роль (“быть подлежащим, будучи первым актантом в данной ситуации” и т.п.)”.

Иными словами, лексическая функция показывает смысловую связь слова с другими словами, способными либо замещать его в тексте при определенных условиях, либо образовывать с ними фразеологические сочетания. При этом богатое словесное варьирование присуще только небольшому числу смыслов, которые и выделяются в качестве стандартных лексических функций-параметров. Данный вид синонимии, именуемый в литературе как ЛФ-синонимия, имеет следующие особенности:

– глубинным синтаксическим структурам (ГСС) сравниваемых высказываний соответствуют одни и те же (или эквивалентные) семантические представления (СемП) [62, с. 32];

– в семантическом графе (СГ) СемП выделяются подграфы (пучки) и каждому подграфу СГ будет соответствовать свое поддерево ГСС каждого из сравниваемых высказываний;

– существует как минимум один подграф СГ, который будет по-разному отображаться в глубинных синтаксических структурах каждого из сравниваемых высказываний. Иными словами, один и тот же смысл в разных ГСС выражается разными обобщенными лексическими единицами [62, с. 178] рассматриваемого ЕЯ. Но при этом перераспределение смысла между лексемами, как показано в [62, с. 147], сводится к минимуму, а смысловые соотношения между цельными лексическими единицами описываются с помощью аппарата стандартных ЛФ.

Как отмечено в [62, с. 147], в силу регулярности стандартных ЛФ и операций над ними ЛФ-синонимические отношения между ГСС оказываются более регулярными и однотипными, нежели чем произвольные синонимические отношения между ГСС. ЛФ-синонимические отношения между ГСС могут быть описаны с помощью специального исчисления в виде системы правил, которая любой данной ГСС ставила бы в соответствие все другие ГСС, ЛФ-синонимичные с ней. При этом саму задачу установления СЭ можно переформулировать следующим образом.

Задача 1.2. Дано:

Π – множество правил ЛФ-синонимических преобразований;

LR – множество пар ЕЯ-высказываний, между которыми возможно установление синонимии (относительно Π);

$Rap(rule_j)$ – множество условий применимости правила $rule_j \in \Pi$. Для $\forall (Ts_1, Ts_2) \in LR \;\; \forall rap_l \in Rap(rule_j)$ есть совокупность требований к Ts_1 и Ts_2.

Требуется: для произвольной пары Lr_k ЕЯ-высказываний проанализировать условие применимости каждого правила множества Π и выделить образ класса

$rule_j \in \Pi$, на который объект Lr_k наиболее похож. При этом rap_l выступает в качестве прецедента как типичного представителя таксона $rule_j$.

Данная задача является классической задачей распознавания образов [40]. Использование прецедентов при таком подходе позволяет сократить объем памяти, необходимой для хранения текстовых баз данных при рассмотрении текстов как сложных информационных объектов с внутренней структурой [151]. Сказанное, в частности, актуально для поисковых систем [122].

Действительно, для каждого текста необходимо выделить его класс СЭ, который соответствует rap_l. Далее происходит поиск уже внутри данного класса того подкласса, который наиболее соответствует данному тексту и включает тексты, максимально синонимичные заданному. По сути, данные о текстах будут описываться некоторой иерархической структурой, каждый новый текст будет определяться только теми признаками, которые отличают этот текст от других представителей наиболее близкого ему класса. Причем в процессе поступления новых текстов в базу классификационные признаки будут постоянно уточняться уже за рамками лексико-функциональной синонимии. Выделение подклассов СЭ при этом производится согласно постановке *задачи 1.1*, сформулированной нами в начале раздела 1.2. Подклассы СЭ будут соответствовать смысловым оттенкам как отдельных слов, так и высказываний в составе языковых конструкций. Заметим, что элементы множества Ts одной и той же ситуации (1.1) в общем случае могут относиться к различным классам СЭ (относительно различных rap_l).

Формирование знаний, соответствующих лексико-функциональной синонимии относительно Π и текстовой кластеризации относительно описываемых текстами объектов (понятий) и ситуаций, будет рассмотрено в третьей и четвертой главах соответственно. Сейчас же мы остановимся более подробно на механизме установления соответствия высказываниям их смыслов для модели “Смысл⇔Текст” как абстрактной модели языка.

Поскольку в модели “Смысл⇔Текст” смысл рассматривается как инвариант всех синонимических преобразований, то семантику следует рассматривать как совокупность правил преобразований одних конструкций ЕЯ в другие конструкции, эквивалентные им по смыслу. Сам смысл при этом задается с помощью формального языка, включающего помимо грамматического компонента и правил перевода конструкций ЕЯ в выражения на языке смыслов процедуру разрешения проблемы эквивалентности языковых конструкций как на уровне ЕЯ, так и на уровне формализованного описания смыслов [148].

Если ограничить рассмотрение синонимии только ЛФ-синонимией, то, как показано в [151], в роли указанного формального языка будет выступать язык глубинного синтаксиса, а в качестве математического аппарата формального описания СЭ – грамматики деревьев вида:

$$\Gamma = (W_R, V_R, \varphi, \Pi, \Phi), \tag{1.2}$$

именуемые в [151] расширенными универсальными правильными Δ-грамматиками. Здесь W_R есть конечное множество (словарь) пометок на узлах; V_R – конечное множество (словарь) пометок на ветвях деревьев. φ есть отображение множества V_R во множество натуральных чисел, представляемое в матричной форме как

$$\varphi = \begin{pmatrix} a_1 & a_2 & \dots & a_k \\ n_1 & n_2 & \dots & n_k \end{pmatrix}, \tag{1.3}$$

где $V_R = \{a_1, a_2, \dots, a_k\}$, а $\{n_1, n_2, \dots, n_k\}$ – подмножество натуральных чисел. Причем предполагается, что все деревья удовлетворяют следующему ограничению: для $\forall i = 1, \dots, k$ из любого узла дерева выходит не более $\varphi(a_i) = n_i$ ветвей с пометкой a_i. В этом случае также говорят, что дерево является φ-правильным [151].

Компоненты Π и Φ в составе пятёрки (1.2) есть конечные множества правил преобразований деревьев. Применительно к ЛФ-синонимическим преобразованиям глубинных синтаксических структур компонент Π имеет содержательную интерпретацию множества синтаксических, а Φ – множества вспомогательных лексиче-

ских правил преобразований деревьев глубинного синтаксиса. Множество V_R здесь есть множество типов глубинно-синтаксических отношений. В рамках подхода “Смысл⇔Текст” рассматриваются шесть типов глубинных синтаксических связей, поэтому $V_R = \{1,2,3,4,5,6\}$. Матрица (1.3) отражает характер ограничений на ветвление в реальных ГСС: $\varphi = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 1 & 1 & 1 & 4 & 1 \end{pmatrix}$. Множество пометок на узлах есть множество характеризованных обобщенных лексем: $W_R = W_{RL} \cup W_{LF} \cup W_{ID} \cup W_{FL}$, где W_{RL} – реальные лексемы языка, W_{LF} – символьные обозначения лексических функций. $W_{FL} = \{Q\}$, Q есть символ фиктивной (пустой) лексемы, которая служит для обозначения узла, не получающего “вещественного” означающего в реальной фразе, но тем не менее его присутствие в дереве глубинного синтаксиса продиктовано семантическими соображениями (пример – незаполненная смысловая валентность у глагола).

Модель СЭ на основе грамматик вида (1.2), исследование ее свойств в аспекте проблем алгоритмической разрешимости и вычислительной сложности детально обсуждается в [151]. Указанная модель использует разнообразную информацию о каждом слове ЕЯ в виде словоизменительных, словообразовательных, синтаксических, семантических и стилистических характеристик слова, описываемых в толково-комбинаторном словаре (ТКС, [162]). В частности, синтаксические и семантические характеристики используются при описании условий применимости правил множества Π. Актуальными здесь являются проблемы автоматизации накопления и систематизации знаний, представляемых ТКС непосредственно на основе текстовых массивов.

1.4. Анализ формальных понятий как инструмент концептуальной кластеризации

Как отмечал И.А. Мельчук в [62, с. 18–20], в модели языка как преобразователя “Смысл⇔Текст” следует выделить лингвистическую (декларативную)

часть, которая представляет собой множество правил соответствия между смыслами и ЕЯ-текстами, и алгоритмическую (процедурную) часть, реализующую механизм использования указанных соответствий. Причем предполагаются переходы от сложных (получаемых операциями комбинирования) смыслов к столь же сложным текстам (то есть также получаемых посредством комбинирования) и наоборот. Будучи независимыми от конкретной процедуры реализации, правила соответствия между смыслами и текстами предполагают конкретизацию условий их применения на ЕЯ-текстах заданной предметной области. На практике это означает необходимость наличия механизма обучения по прецедентам в рамках указанной составляющей модели “Смысл⇔Текст”.

Как отмечал академик Ю.Д. Апресян [3, с. 335–336], ограничения, накладываемые, в частности, на синонимические преобразования глубинных синтаксических структур, зависят как от особенностей отдельных слов, так и целых пластов лексики. Актуальным здесь является выбор подходящей модели представления знаний о синонимии в совокупности с подходом к их систематизации и упорядочиванию.

Согласно формулировке *задачи 1.2*, а также данному в [62, с. 151] определению условия применимости правила ЛФ-синонимического преобразования, прецедент класса СЭ определяется в первую очередь совокупностью требований к синтаксическим и семантическим свойствам тех лексических единиц, которые участвуют в выполняемой посредством правила замене. При этом информация, связанная с лексемой, включает денотативный и смысловой компоненты.

Определение 1.5. Денотат ЕЯ-слова есть множество сущностей реального мира, которые этим словом могут быть правильно названы. В отличие от референции, денотат является частью значения слова и не зависит от контекста конкретной ситуации употребления ЕЯ.

Понятие смысла слова в целом сходно с понятием смысла высказывания, даваемым *определением 1.3*.

Определение 1.6. Смысл слова определяется как множество отношений вида “денотат – денотат” (именуемых также смысловыми отношениями), существующих между данным словом и другими словами в заданном естественном языке.

В логике различие между смыслом и денотатом определяется с помощью экстенсионала и интенсионала.

Определение 1.7. Экстенсионал (объем понятия) есть класс сущностей, именуемых заданным словом.

Определение 1.8. Интенсионал (содержание понятия) есть множество признаков, определяющих класс сущностей из экстенсионала.

Как следует из *определений 1.7* и *1.8*, экстенсионал соответствует денотату, интенсионал – смыслу слова.

Само описание слова с точки зрения объема и содержания понятия, обозначаемого словом, составляет основу кластеризации слов, наиболее естественно реализуемой методами анализа формальных понятий (АФП) [28, 140, 141, 153, 154, 158, 159, 179, 184, 185].

Определение 1.9. АФП – это метод анализа данных, основанный на математической теории решеток [4]. Основой АФП является доказанная Г. Биркгофом теорема [4] о том, что для любого бинарного отношения можно построить полную решетку.

При использовании данного метода некоторая исследуемая область знаний описывается в терминах набора объектов и признаков (атрибутов), затем вводится описание формального контекста. Далее для заданного контекста формируется множество формальных понятий, и строится решетка, которая может быть визуально отображена диаграммой линий. Формализация понятий и их последующий анализ с помощью решетки позволяют оперировать данными на семантическом уровне без потери или недопустимого упрощения объектов и их признаков.

Классификация объектов и результаты анализа данных с помощью анализа формальных понятий могут быть интерпретированы исследователем для заданной предметной области.

Приведем используемые далее основные определения из теории АФП.

Пусть G – множество объектов, M – множество признаков для объектов из G. Имеем также бинарное отношение $I \subseteq G \times M$. Если $g \in G$ и $m \in M$, то gIm имеет место тогда и только тогда, когда g обладает признаком m.

Определение 1.10. Тройка $K = (G, M, I)$ называется формальным контекстом. При этом для произвольных $A \subseteq G$ и $B \subseteq M$ вводится пара отображений: $A' = \{m \in M \mid \forall g \in A : g\,\mathrm{Im}\}$ и $B' = \{g \in G \mid \forall m \in B : g\,\mathrm{Im}\}$.

Определение 1.11. Пара множеств (A, B), таких что $A \subseteq G$, $B \subseteq M$ и $A' = B$, $B' = A$, называется формальным понятием (ФП) с объемом A и содержанием B.

Определение 1.12. ФП (A_1, B_1) называют подпонятием для ФП (A_2, B_2), если $A_1 \subseteq A_2$. При этом (A_2, B_2) называют суперпонятием для ФП (A_1, B_1) (обозначается как $(A_1, B_1) \leq (A_2, B_2)$). Отношение $\leq$ будем называть отношением порядка для формальных понятий.

Определение 1.13. Формальные понятия C_1 и C_2 считаются сравнимыми, если либо $C_1 \leq C_2$, либо $C_2 \leq C_1$. В противном случае эти ФП называют несравнимыми.

Определение 1.14. Множество всех ФП контекста $K = (G, M, I)$ вместе с заданным на нем отношением $\leq$ обозначают $\Re(G, M, I)$ и называют *решеткой формальных понятий*.

Определение 1.15. Подмножество множества формальных понятий, в котором каждые два элемента являются сравнимыми, называют *цепочкой*, а если каждые два элемента являются несравнимыми, называют *антицепочкой*.

Определение 1.16. Под областью в решетке ФП понимается набор формальных понятий, связанных отношением порядка с одним наибольшим общим подпонятием (НОПП) и/или одним наименьшим общим суперпонятием (НОСП). В роли НОПП может выступать наименьшее ФП в решетке, а в роли НОСП – вершинное ФП.

Определение 1.17. ФП C_2 называется соседним по отношению к ФП C_1 в решетке $\Re$, если они имеют НОСП, отличное от вершинного ФП в этой решетке.

Замечание. АФП по определению есть инструмент концептуальной кластеризации, так как $\forall (A,B) \in \Re$ есть класс с заданной интерпретацией в виде содержания – множества B.

Далее в работе мы покажем, каким образом с помощью АФП выполняется формирование и классификация условий применимости ЛФ-синонимических преобразований в *задаче 1.2*, решается задача текстовой кластеризации, включающая *задачу 1.1* в качестве подзадачи.

Кроме того, посредством АФП реализуется механизм согласования различных уровней синонимии в естественном языке, и определяются меры схожести ситуаций языкового употребления.

Выводы

Анализируя задачу текстовой кластеризации, описанную нами в общих чертах в разделе 1.3, можно констатировать, что:

– ситуация языкового употребления как единица формализованного описания семантики языка может служить источником знаний как о лексико-функциональной синонимии, представляющей верхний уровень иерархии знаний о синонимии, так и о произвольных случаях СЭ в ЕЯ;

– АФП представляет собой инструмент формирования и кластеризации понятий, с которыми могут быть отождествлены классы СЭ;

– решетка формальных понятий является удобным формализмом для представления текстовой информации в сжатом виде [57], сами тексты при этом объединяются в классы по сходству признаков сочетаемости слов относительно контекстов, определяемых ситуациями языкового употребления.

При этом задача иерархизации знаний о синонимии в заданном ЕЯ сводится к совокупности следующих подзадач, решаемых далее в третьей, четвёртой и пятой главах настоящей диссертационной работы:

– выделение и кластеризация отношений в рамках ситуации языкового употребления, представляемой тройкой (1.1);

– формирование прецедентов для ситуаций ЛФ-синонимии в соответствии со сформулированной нами *задачей 1.2* на основе полученных отношений;

– введение оценок схожести для ситуаций языкового употребления на основе формализованного представления знаний о синонимии в заданном предметно-ориентированном подмножестве естественного языка.

Язык глубинных синтаксических структур как средство описания синтаксических и лексических правил синонимического перифразирования при всех своих несомненных достоинствах обладает и одним существенным недостатком, а именно: на указанном уровне текст представляется пофразно, каждая фраза соответствует простому распространенному предложению. Отсюда возникает проблема полноты описания смысла при формировании прецедентов классов СЭ для *задачи 1.2*. Если одна из форм описания рассматриваемой ситуации действительности представлена ЕЯ-высказыванием, состоящим более чем из одной фразы, то в соответствии с *задачей 1.1* выделение множеств ситуаций, описываемых текстом, и объектов, значимых в этих ситуациях, должно производиться только на основе анализа сходств классов СЭ всех фраз, составляющих высказывание.

Решению проблемы полноты представления смысловой информации деревьями глубинного синтаксиса при формировании прецедентов классов СЭ на уровне абстрактной лексики посвящается вторая глава диссертационной работы.

Глава 2
ТЕОРЕТИЧЕСКИЕ ОСНОВЫ РАСПОЗНАВАНИЯ СВЕРХФРАЗОВЫХ ЕДИНСТВ НА УРОВНЕ ГЛУБИННОГО СИНТАКСИСА

На основе полученной в первой главе формальной концептуальной модели семантической эквивалентности в настоящей главе исследуется проблема полноты представления смысла при формирования прецедентов ситуаций синонимии для уровня абстрактной лексики. При этом содержательную основу сжатия смысловой информации составляют сверхфразовые единства на уровне глубинного синтаксиса. Ставится и решается задача их выделения и формального представления для ЕЯ-высказываний, состоящих более чем из одного простого распространенного предложения. Рассмотрены вопросы эффективного использования знаний уровня глубинного синтаксиса при решении указанной задачи. Основные результаты главы опубликованы в [35, 67, 71, 78, 79, 81, 82, 84, 87, 88, 146, 170, 176].

2.1. Концептуальная модель процесса распознавания взаимной дополняемости фраз в сравниваемых по смыслу высказываниях естественного языка

Целью настоящего раздела является описание на функциональном уровне задачи увеличения полноты описания смысла в формальном образе ЕЯ-текста при установлении его эквивалентности смысловому эталону-образцу с использованием определяемых в этом же разделе основополагающих понятий и терминологии.

Прежде чем описывать процесс построения формальных семантических образов сверхфразовых единств, введем ряд понятий, характеризующих рассмотренный в [151] процесс установления семантической эквивалентности текстов в рамках подхода “Смысл⇔Текст”. Будем считать, что в общем случае сравниваемые ЕЯ-тексты состоят из различного числа фраз, а одна фраза соответствует простому распространенному ЕЯ-предложению.

Во-первых, предложенная в [151] модель семантической эквивалентности работает с совокупностями деревьев глубинно-синтаксических структур фраз, к каждому из этих деревьев должно быть применено одно или несколько правил синонимического перифразирования. В [151] исследуется алгоритмическая сложность задачи применения правила к помеченному дереву без рассмотрения анализируемых входом правила компонент этого дерева. Подобное рассмотрение правил синонимических преобразований не позволяет говорить о необходимых и достаточных признаках синонимии двух фраз по анализу применимости к ним правил синонимических преобразований и, как следствие, целесообразности синонимических трансформаций того или иного типа, что позволило бы в значительной степени сократить перебор в задаче установления семантической эквивалентности ЕЯ-текстов. Поэтому рассмотрим более подробно процесс применения правила расширенной лексико-синтаксической Δ-грамматики вида (1.2) к некоторому дереву Tr_i с пометками на ветвях и в узлах (в содержательной интерпретации – применение лексического правила с обслуживающим его синтаксическим правилом синонимических замен к дереву глубинного синтаксиса).

Согласно принятому в теории языка как преобразователя “Смысл⇔Текст” делению правил синонимических преобразований на лексические и синтаксические и взаимозависимостью применений правил указанных типов, в процессе применения правила следует выделить:

– определение поддерева, заменяемого лексическим правилом + первым из обслуживающих его синтаксических правил с фиксацией номеров правил;

– определение ключевого слова (C_0) комплекса лексических единиц, заменяемых лексическим правилом [62, с. 150].

Определение 2.1. Под лексической синонимической конструкцией (ЛСК) следует понимать комплекс лексических единиц $wr_k \in W_R$ и связей $vr_j \in V_R$ между ними, замена которого описывается некоторым правилом $\phi_i \in \Phi$ Δ-грамматики вида (1.2). Каждой ЛСК соответствует свое ключевое слово C_0, либо непосредст-

венно входящее в нее, либо выраженное в значениях ЛФ от C_0 (лексических коррелятов C_0) в комплексе составляющих ЛСК лексических единиц.

Содержательно лексические единицы в составе ЛСК – это обобщенные лексемы и их лексические корреляты, а связи между лексическими единицами соответствуют глубинно-синтаксическим отношениям. Сами замены соответствуют лексическим правилом синонимического перифразирования.

На основе полученного определения ЛСК сформулируем необходимые и достаточные условия ЛФ-синонимии глубинных синтаксических структур. Согласно определению сравниваемым ГСС здесь соответствуют эквивалентные СемП, выраженные одним и тем же семантическим графом. Причем элементы этого СГ группируются в разных ГСС в одинаковые пучки, выраженные разными лексическими единицами, соотношения между которыми описываются с помощью аппарата стандартных ЛФ.

Определение 2.2 (необходимое условие ЛФ-синонимии ГСС). Будем считать, что ГСС фраз F_1 и F_2: $F_1 \neq F_2$ удовлетворяют необходимому, но не достаточному условию ЛФ-синонимии, если их ЛСК относится к одному и тому же ключевому слову C_0.

Данное условие позволяет определить возможность наличия некоторой ГСС в множестве деревьев, получаемом для заданной ГСС с применением лексических правил синонимических замен без построения этого множества. Действительно, если для рассматриваемых ГСС их ЛСК относятся к разным ключевым словам, то получение на основе некоторой ГСС дерева глубинного синтаксиса, эквивалентного заданному, возможно только посредством чисто синтаксических замен, не требующих замен лексики.

Замечание. Поскольку лексические замены ведутся относительно определенного ключевого слова C_0, то невыполнение необходимого условия ЛФ-синонимии для одних ЛСК в сравниваемых деревьях не означает невозможности отношения ЛФ-синонимии между рассматриваемыми деревьями, поскольку к одному и тому же

дереву глубинного синтаксиса может быть применено несколько лексических правил перифразирования, что позволяет говорить об относительности ЛФ-синонимии.

Представим вход правила ЛФ-синонимической замены как описание поддерева, заменяемого первым из обслуживающих данное лексическое преобразование синтаксических правил, внутри которого содержится описание поддерева, заменяемого лексическим правилом. Тогда определение возможности применения синонимических преобразований из заданного множества Π есть определение применимости каждого правила $rule_j \in \Pi$, с выделением ключевого слова ЛСК и представлением результата в виде списка пар:

$$\left\{\left(rule_j, C_0(j)\right): j = 1,\ldots,|\Pi|\right\}. \qquad (2.1)$$

Сама Δ-грамматика (1.2) при этом может быть редуцирована до четвёрки $\Gamma = (W_R, V_R, \varphi, \Pi)$, именуемой в [14] лексико-синтаксической Δ-грамматикой, а вместо правил $\phi_i \in \Phi$ в *определении 2.1* следует рассматривать правила $rule_j \in \Pi$. Множество Π здесь, как показано в [14], конечно, что также очевидным образом вытекает из конечности множеств Π и Φ в составе пятёрки (1.2).

Если для некоторой ГСС, входящей в множество ГСС смыслового описания одного высказывания, ключевое слово C_0 одного из элементов списка (2.1) совпадает с ключевым словом одного из элементов аналогичного списка у некоторой ГСС из смыслового описания второго, “эталонного” высказывания, то дальнейшие действия по установлению эквивалентности указанных ГСС включают в себя:

– построение некоторой последовательности лексических преобразований, приводящих поддеревья исходных ГСС, заменяемые лексическими правилами + первыми из обслуживающих их синтаксических правил, к виду с одинаковой ЛСК;

– сравнение путем наложения, начиная с вершины, при совмещении одноименных стрелок преобразованных ГСС на предмет эквивалентности.

Определим понятие эквивалентности (равенства) ГСС.

Определение 2.3. Помеченные деревья Tr_1 и Tr_2 (в содержательной интерпретации – деревья глубинного синтаксиса) являются эквивалентными (равными, тождественными), если они изоморфны [14] таким образом, что для всякого узла α дерева Tr_1 его образ $f(\alpha)$ в дереве Tr_2 имеет одинаковую с ним пометку.

Как показано в [14], применение некоторого преобразования в Δ-грамматике сводится к последовательному выполнению:

– декомпозиции [68] исходного дерева с выделением заменяемого поддерева и расстановкой композиционных меток, обозначающих выделенные узлы;

– композиции [14] дерева верхнего контекста заменяемого дерева, заменяющего дерева и деревьев нижнего контекста в соответствии с порядком, задаваемым композиционными метками.

Таким образом, для установления эквивалентности деревьев глубинного синтаксиса Tr_1 и Tr_2, приведенных к виду с одинаковой ЛСК, необходимо вначале выполнить сравнение замененных поддеревьев, включающих ЛСК, а затем деревьев верхнего и нижнего контекста замененных поддеревьев. Последние в результате последовательности трансформаций остаются без изменений.

Показанное свойство ЛФ-синонимических преобразований позволяет рассматривать ЛФ-синонимию ГСС при задании их ЛСК относительно одного и того же ключевого слова C_0 как частный случай семантического повтора на основе значений лексических функций самостоятельных лексем [114]. При этом ЛСК рассматриваются в качестве элементов повтора, представляя собой комбинации значений лексических функций заданного ключевого слова, связанных отношениями глубинного синтаксиса.

Определим формально взаимную дополняемость глубинных синтаксических представлений при задании их ЛСК относительно одного и того же ключевого слова.

Определение 2.4. Будем считать, что отвечающие необходимому условию ЛФ-синонимии (в соответствии с *определением 2.2*) деревья Tr_1 и Tr_2 удовле-

творяют необходимому (но не достаточному!) условию смысловой взаимной дополняемости, если существует последовательность ЛФ-синонимических преобразований, приводящих Tr_1 и Tr_2 к виду с одинаковой ЛСК.

Для дальнейшего изложения введем в рассмотрение семантические словоизменительные характеристики лексем, представляемых в узлах дерева глубинного синтаксиса. Согласно данному в [62, с. 144] определению к таковым относятся число для существительных и вид, время, наклонение – для глагола.

Определение 2.5. Будем считать, что удовлетворяющие (согласно *определению 2.4*) необходимому условию взаимной дополняемости и приведенные к виду с одинаковой ЛСК деревья Tr_1 и Tr_2 взаимно дополняют друг друга, если они изоморфны так, что для всякого узла α дерева Tr_1 его образ $f(\alpha)$ в дереве Tr_2:

– либо содержит информацию об одной и той же характеризованной обобщенной лексеме [62, с. 144] данного ЕЯ, не являющейся нулевой (фиктивной) лексемой [62, с. 143];

– либо представляет обозначенную символом Q фиктивную лексему с теми же семантическими словоизменительными характеристиками, что и ненулевая характеризованная обобщенная лексема, информация о которой содержится в узле α;

– либо представляет ненулевую характеризованную обобщенную лексему с теми же семантическими словоизменительными характеристиками, что и фиктивная лексема, информация о которой содержится в узле α.

Следствие. Рассматриваемая *определением 2.3* эквивалентность (равенство) ГСС является частным случаем взаимной дополняемости деревьев глубинного синтаксиса.

Замечание. В реальных ЕЯ-текстах достаточно много случаев, когда удовлетворяющие (согласно *определению 2.4*) необходимому условию взаимной дополняемости и приведенные к одинаковой ЛСК деревья Tr_1 и Tr_2 не могут взаимно дополнять друг друга. Причина кроется в том, что существует как минимум один

узел α дерева Tr_1, образ $f(\alpha)$ которого в дереве Tr_2 содержит информацию о ненулевой характеризованной обобщенной лексеме с теми же семантическими словоизменительными характеристиками, что и отличная от нее ненулевая характеризованная обобщенная лексема, представляемая узлом α. Будем считать, что в этом случае Tr_1 и Tr_2 имеют *ложную взаимную дополняемость*.

Таким образом, увеличения полноты смыслового описания текста, сравниваемого с эталоном на уровне глубинного синтаксиса, можно достичь суммированием глубинных синтаксических структур, взаимно дополняющих друг друга, путем наложения при совмещении одноименных стрелок с “заполнением мест”, соответствующих фиктивным (нулевым) лексемам. При этом исходные ГСС, сведенные к единой (“суммарной”) ГСС, исключаются из смыслового описания анализируемого текста. В содержательной лингвистической интерпретации это означает для анализируемого текста построение образов сверхфразовых единств [118] на уровне глубинного синтаксиса.

Определение 2.6. Формальным образом сверхфразового единства (в дальнейшем – сверхфразовым единством) на глубинном синтаксическом уровне представления смысловых образов фраз будем называть дерево глубинного синтаксиса, полученное суммированием глубинных синтаксических структур, взаимно дополняющих друг друга по *определению 2.5*, путем наложения при совмещении одноименных стрелок с “заполнением мест”, соответствующих фиктивным (нулевым) лексемам.

С учетом распознавания сверхфразовых единств, введенного в модель семантической эквивалентности, функционирование механизма установления семантической эквивалентности высказываний будет представляться следующей концептуальной моделью, полученной расширением соответствующей модели, представленной в [148, 151].

Для заданного ЕЯ Y вводится в рассмотрение язык смыслов Y_S. Как было показано в разделе 1.3, при рассмотрении смысла как инварианта синонимиче-

ских преобразований в качестве Y_S будет выступать язык глубинного синтаксиса. Сам язык Y_S при этом представляется упорядоченной пятеркой:

$$Y_S = \langle L_S, \Gamma_S, \Pi_S, Q_S, U_S \rangle, \tag{2.2}$$

где L_S – лексика языка Y_S;

Γ_S – синтаксис языка Y_S;

Π_S – процедура установления соответствий между фразами языков Y и Y_S;

Q_S – процедура, с помощью которой решается проблема эквивалентности в языке Y_S;

U_S – процедура, преобразующая смысловое представление анализируемого текста на основе учета описанных выше семантических повторов.

Процедура Q_S содержит допустимые L_S и Γ_S лексические и синтаксические правила преобразований эквивалентных смысловых образов друг в друга (фактически – правила ЛФ-синонимических преобразований ГСС). Компонента U_S описывает приведение фраз, связанных по смыслу в языке Y_S (по мнению носителя языка Y), к формальному представлению, допускающему нахождение искомого суммарного смысла (в содержательной интерпретации – к виду с одинаковой ЛСК). Кроме того, в составе U_S содержатся правила построения единого смыслового образа для “приведенных” фраз языка Y_S. Исходя из вышесказанного, представим компоненту U_S упорядоченной двойкой:

$$U_S = \langle Q_U, S_U \rangle, \tag{2.3}$$

где Q_U – процедура приведения фраз в Y_S, связанных по смыслу (по мнению носителя языка Y), к формальному представлению, допускающему нахождение искомого суммарного смысла (т. е. к виду с одинаковой ЛСК). Процедура Q_U использует допустимые L_S и Γ_S лексические и синтаксические преобразования с наложением необходимых ограничений;

S_U – процедура, содержащая правила построения единого смыслового образа для “приведенных” фраз из Y_S (суммарного смысла в языке Y_S).

Язык Y_S обладает следующими свойствами, актуальными для решения задачи распознавания семантических повторов в сравниваемом с эталоном ЕЯ-тексте:

1) если фразы F_1 и F_2 языка Y (по мнению его носителя) эквивалентны по смыслу, то с помощью Π_S обе эти фразы либо переводятся в одну и ту же фразу языка Y_S, либо переводятся в две фразы Φ_1 и Φ_2, но такие, что Φ_1 и Φ_2 эквивалентны в Y_S;

2) если фразы F_1 и F_2 языка Y (по мнению его носителя) взаимно дополняют друг друга по смыслу, то полученные с помощью процедуры Π_S образы Φ_1 и Φ_2 этих фраз в языке Y_S процедурой Q_U сводятся к виду, допускающему нахождение искомого суммарного смысла, а затем посредством процедуры S_U переводятся в одну фразу Φ языка Y_S, соответствующую образу суммарного смысла;

3) если фразам F_1 и F_2 языка Y соответствуют полученные с помощью процедуры Π_S фразы Φ_1 и Φ_2 языка Y_S, сводимые процедурой Q_U к представлению, допускающему нахождение искомого суммарного смысла, но не сводимые с помощью процедуры S_U в единую фразу языка Y_S, то фразы F_1 и F_2 следует считать фразами с ложной смысловой взаимной дополняемостью.

Кроме того, предполагается наличие необходимых, но не достаточных признаков наличия семантической связи между фразами из Y на основе анализа их образов в Y_S (см. *определения 2.2* и *2.4*). В силу родственной природы задач установления семантической эквивалентности и распознавания семантических повторов указанные признаки берутся в качестве необходимых, но не достаточных признаков эквивалентности фраз. Для анализа возможностей использования та-

ких признаков рассмотрим более подробно структуру множества фраз постулируемого языка смыслов Y_S.

Следует отметить, что введение в рассмотрение смысловых повторов подразумевает двойное разбиение множества Φ_S фраз языка Y_S. С одной стороны, указанное множество разбивается на непересекающиеся подмножества:

$$\Phi_S = \Phi_{S1} \cup \Phi_{S2} \cup \ldots \cup \Phi_{Sk}, \qquad (2.4)$$

в каждом из которых фразы эквивалентны между собой, но ни одна фраза $\Phi_1 \in \Phi_{Si}$ для $i = 1, \ldots, k$ не будет эквивалентна ни одной другой фразе $\Phi_2 \in \Phi_{Sj}$ для $j = 1, \ldots, k$, если $i \neq j$.

С другой стороны, то же самое множество фраз можно разбить на непересекающиеся множества Φ_{LSCi}, имеющих ЛСК, задаваемые каждое относительно своего ключевого слова. При этом особым подмножеством множества Φ_S будет множество Φ_{SYNT} фраз языка Y_S, для которых не может быть определена ЛСК (могут быть применены только синтаксические трансформации, допустимые Γ_S):

$$\Phi_S = \Phi_{LSC1} \cup \Phi_{LSC2} \cup \ldots \cup \Phi_{LSCl} \cup \Phi_{SYNT}. \qquad (2.5)$$

При этом Φ_{Si}, Φ_{LSCj} и Φ_{SYNT} связаны друг с другом следующим образом. Каждое из Φ_{Si} может включать в себя элементы разных Φ_{LSCj}, а также элементы Φ_{SYNT}. Иначе говоря, каждое Φ_{Si} есть множество, которое может включать подмножества нескольких множеств Φ_{LSCj} плюс некоторое подмножество множества Φ_{SYNT}. Содержательно это соответствует принципу относительности выделения ЛСК: к одной и той же фразе Φ могут быть применены несколько правил преобразований, причем каждое относительно своей ЛСК. Более того, что особенно важно для построения процедуры S_U, в каждом множестве Φ_{LSCj} выделяется два подмножества:

– множество пар фраз, взаимно дополняющих друг друга по смыслу: $\Phi\Pi_j = \{(\Phi_1, \Phi_2) : U_S(\Phi_1, \Phi_2, \Phi) = true\} \subset \Phi_{LSCj} \times \Phi_{LSCj}$, подмножеством которого является множество пар фраз, эквивалентных между собой;

– множество пар фраз с ложной взаимной дополняемостью.

Можно показать, что если $\{(\Phi_1, \Phi_2), (\Phi_3, \Phi_4)\} \subset \Phi\Pi_j$, то из этого вовсе не следует того, что $\{(\Phi_1, \Phi_3), (\Phi_2, \Phi_4), (\Phi_1, \Phi_4), (\Phi_2, \Phi_3)\} \subset \Phi\Pi_j$.

Таким образом, модель (2.2) отвечает выдвинутому в главе 1 требованию согласования различных уровней синонимии между собой. Тем не менее предложенная модель является концептуальной, не имея в своем составе средств описания модели и аппарата манипулирования данными в плане:

– описания механизма применения определенных в процедуре Q_S лексических и синтаксических преобразований фраз множества Φ_S;

– описания процедуры U_S;

– описания взаимодействия процедуры Q_S с процедурой U_S в процессе установления эквивалентности в языке Y_S.

Указанные задачи предполагают построение и исследование модели процесса приведения глубинной синтаксической структуры к некоторому заданному виду. Такая модель ориентирована на формализованное описание входа/выхода правила как информационного элемента и предусматривает различные ситуации его активизации. Этим вопросам посвящаются три последующие раздела настоящей главы.

2.2. Построение системы целевых выводов в Δ-грамматике

В данном разделе решается задача приведения ГСС фраз к виду, допускающему нахождение суммарного смысла (к виду с одинаковой ЛСК). Рассматривается построение системы целевых выводов в Δ-грамматике, реализуемое процедурой Q_U в составе концептуальной модели (2.2)–(2.3).

Решение задачи получения на основе исходной ГСС другой ГСС, удовлетворяющей некоторым функциональным требованиям, при использовании заданной системы правил синонимического преобразования помеченных деревьев, требует исследования динамики функционирования Δ-грамматики, которая моделирует указанную систему. С этой целью в настоящем разделе мы рассмотрим модель отдельного правила Δ-грамматики для последующего описания структуры информационного пространства, соответствующего системе таких правил.

В настоящей работе, говоря о правилах Δ-грамматики, мы имеем в виду подмножество произвольных элементарных преобразований [14, 151], которыми моделируются глубинные синтаксические преобразования конкретного рассматриваемого ЕЯ. При дальнейшем изложении, говоря о правилах Δ-грамматики, мы будем подразумевать произвольные элементарные преобразования, опуская этот термин. Будем рассматривать соответствующие правилам Δ-грамматики переходы между помеченными деревьями (в содержательной интерпретации – переходы от одной ГСС к другой, ЛФ-синонимичной с ней) как односторонние. Если же некоторое правило выполняется в обе стороны, то ему будут соответствовать два возможных перехода, каждый из них выполняется в своем направлении. Следует отметить, что в отличие от динамических информационных структур, используемых для построения интерактивных графических систем [117, 144, 147], связи между входами и выходами правил как информационными элементами задаются изначально и не могут быть изменены в процессе функционирования системы.

Рассмотрим работу некоторого правила $rule_j \in \Pi$. В общем случае здесь следует выделить:

- состояние, соответствующее заменяемому дереву Tio_1;
- состояние, соответствующее заменяющему дереву Tio_2;
- условие rap_l срабатывания правила $rule_j$ для Tio_1 и Tio_2.

Иными словами, мы имеем простейший случай задачи достижимости ЛСК с заданными свойствами на информационном пространстве, заданном входами и выходами правил $rule_j \in \Pi$. Решение такой задачи есть ответ на ряд вопросов:

– удовлетворяет ли исходное дерево требованиям входного дерева Tio_1 рассматриваемого правила $rule_j$;

– удовлетворяет ли целевое дерево требованиям выходного дерева Tio_2 правила $rule_j$;

– возможен ли переход от Tio_1 к Tio_2 с учетом информационного наполнения исходного и целевого деревьев в совокупности с характером условия rap_l.

Более общий случай задачи достижимости ЛСК с заданными свойствами отличается от описанного простейшего тем, что:

– рассматриваются входы и выходы не одного, а разных правил $rule_1 \in \Pi$ и $rule_2 \in \Pi$, $rule_1 \neq rule_2$;

– исследуется возможность не одного, а последовательности переходов от Tio_1 к Tio_2.

Условие rap_l применимости правила $rule_j \in \Pi$ содержит список требований к узлам и ветвям входного и выходного дерева, представляя собой формальное описание допустимости перехода из состояния Tio_1 в состояние Tio_2. Учитывая особенности реальных систем перифразирования, наиболее целесообразно для каждого $rule_j \in \Pi$ рассматривать именно множество условий его применимости (согласно постановке *задачи 1.2*), из которых для срабатывания правила должно выполниться как минимум одно.

Переход из состояния Tio_1 в состояние Tio_2 возможен при условии свершения совокупности событий, соответствующих обнаружению в глубинном синтаксическом дереве на определенных позициях узлов с заданными характеристиками.

В отличие от динамических информационных структур, используемых для построения интерактивных графических систем, в рассматриваемой задаче изменение состояния системы может быть вызвано не только отдельным событием, но и их совокупностью, причем в большинстве правил имеет место именно совокупность событий.

Опишем формально совокупности событий, определение которых используется компонентой Rap. С учетом вышесказанного, множество Rap есть множество совокупностей событий из множества X всех событий, допустимых всеми системами правил множества Π. В содержательной интерпретации каждое $x_i \in X$ есть обнаружение в глубинном синтаксическом дереве на определенной позиции узла с некоторыми характеристиками и ему соответствует значение либо *“true”* (обнаружение), либо *“false”* (необнаружение). Применение правила $rule_j \in \Pi$ рассматривается как переход из состояния Tio_1 в состояние Tio_2, который будет возможен, если существует совокупность событий $(x_1, x_2, \ldots, x_n)$ такая, что $x_1 \wedge x_2 \wedge \ldots \wedge x_n = true$, и существует условие применимости $rap_l \in Rap$:

$$rap_l = x_1 \wedge x_2 \wedge \ldots \wedge x_n . \qquad (2.6)$$

Правило $rule_j \in \Pi$ может быть применено к дереву Tio_1, если $\vee_{l=1}^{m} rap_l = true$, где $m = |Rap|$. Обозначим $\vee_{l=1}^{m} rap_l$ для дальнейшего использования как r_{12}. Условие r_{12} следует интерпретировать как “определение события, разрешающего переход от Tio_1 к Tio_2”.

Применение правила $rule_j \in \Pi$ сводится к выполнению перехода:

$$rule_j(r_{12}) : Tio_1 \xrightarrow{rule_j(r_{12})} Tio_2 . \qquad (2.7)$$

Предложенное описание правила Δ-грамматики естественным образом согласуется с математическим аппаратом сетей Петри [53, 104]. Отдельному правилу соответствует элементарная сеть Петри вида

$$N = \{P, T, F, H, M_0\}. \qquad (2.8)$$

При этом множество состояний правила есть множество P позиций (мест) сети: $P=\{p_1, p_2\}$, где $p_1 \Leftrightarrow Tio_1$, а $p_2 \Leftrightarrow Tio_2$. Множество возможных переходов T сети представлено единственным переходом из состояния Tio_1 в состояние Tio_2: $t = rule_j(r_{12})$: $p_1 \xrightarrow{t} p_2$. Компоненты F и H представляют отображения, задаваемые матрицами инцидентности $F : P \times T \to \{0,1\}$ и $H : T \times P \to \{0,1\}$, соответственно. Согласно данному в [104] определению, для любой $p_i \in P$ $F(p_i, t)=1$, если p_i является входной позицией перехода t. Аналогично $H(t, p_i)=1$, если p_i – выходная позиция перехода t. Для сети вида (2.8) имеем: $F(p_1, t)=1$, $F(p_2, t)=0$, $H(t, p_1)=0$, $H(t, p_2)=1$. Число допустимых разметок сети здесь равно двум. В рассматриваемой модели одновременно активным может быть только один информационный элемент, соответствующий либо входу, либо выходу правила. Поскольку множество мест в сети изначально упорядочено (порядок соответствует состояниям моделируемого правила), каждая из допустимых разметок может быть представлена в виде двоичного вектора длины, равной числу позиций, то есть 2. Начальной маркировке соответствует вектор $M_0 = (1,0)$, второй из допустимых маркировок – вектор $M = (0,1)$. Вторая разметка является тупиковой.

Ввиду того, что множество правил Π используется компьютерной программой, а не пользователем-человеком, следует формально определить функцию активизации входа правила, являющейся функцией активизации (запуска или начальной маркировки [104]) сети Петри. Указанная функция формально определяется как логическая функция, выдающая либо “*true*”, если анализируемое дерево глубинного синтаксиса функционально соответствует входному дереву Tio_1 правила $rule_j \in \Pi$, либо “*false*” в противном случае. По значению этой функции происходит (в случае “*true*”) либо не происходит (в случае “*false*”) начальная маркировка $M_0 = (1,0)$ рассматриваемой сети Петри.

Рассмотрим ограничения, накладываемые на классический аппарат сетей Петри, применительно к моделированию отдельного правила Δ-грамматики.

Во-первых, правило $rule_j \in \Pi$ моделируется элементарной сетью Петри, в которой число фишек (маркеров) в каждой позиции не превышает 1. Следует отметить, что это ограничение накладывается из содержательных особенностей моделируемого объекта, а не является свойством топологии сети. В содержательном плане это означает, что за один проход (одно срабатывание правила Δ-грамматики) не может быть обработано более одного дерева.

Во-вторых, введена функция, определяющая возможность срабатывания перехода при выполнении определенного в классическом аппарате сетей Петри условия срабатывания (наличие фишек в каждой из входных позиций) путем анализа совокупности событий, сопутствующей активизации позиции, инцидентной данному переходу. Содержательно такое ограничение ведет к появлению тупиковых разметок второго рода: условие активизации инцидентных переходу позиций выполнено, но переход сработать не может, поскольку функция $t = rule_j(r_{12})$ активизации перехода $p_1 \xrightarrow{t} p_2$, соответствующая условиям применимости рассматриваемого правила $rule_j \in \Pi$, выдает “*false*”.

Множество представленных элементарными сетями Петри правил Δ-грамматики можно рассматривать как множество исходных объектов-примитивов для построения в терминах ограниченных сетей Петри [53, 104] информационной модели системы правил некоторого подмножества множества Π рассматриваемой Δ-грамматики с определением структурных взаимосвязей между ними. При этом сама система правил формируется следующим образом: для каждой пары правил $\{rule_1, rule_2\} \subset \Pi$, $rule_1 \neq rule_2$, входящих в систему, обязательным является выполнение следующего условия: либо вход правила $rule_2$ является выходом для $rule_1$, либо наоборот, вход у $rule_1$ является выходом для правила $rule_2$.

Следует отметить, что для любой Δ-грамматики такие системы могут быть определены изначально и не обладают свойством динамичности: связи входов и выходов правил детерминированы, а та роль, которую выполняет пользователь-человек в моделях мультимедийных приложений [147], принадлежит машине, работающей по жестко заданной логике, определяемой системой перифразирования рассматриваемого ЕЯ, что исключает фактор случайности.

Рассмотрим динамику функционирования совокупности правил из множества $П$, образующих систему, для случая, когда одновременно активизируются входы у двух различных правил. Подобным образом функционирует система перифразирования ЕЯ при приведении ГСС фраз к целевому виду.

Отметим, что для построения практически значимой модели системы синонимического перифразирования недостаточно простого описания совокупности переходов от одного ЛФ-синонимичного представления к другому. Простое перечисление правил, условий их применения, обслуживающих правил не учитывает:

– преобразования, выполняемые согласно требованиям моделей управления (МУ) предикатных слов, указываемым в их словарных статьях;

– возможность определения по заданной системе правил Δ-грамматики выводимости ГСС с заданными свойствами.

Получение дерева с требуемыми свойствами при распознавании семантических повторов на уровне глубинного синтаксиса означает поиск по совокупности правил заданной Δ-грамматики (с учетом приоритета каждого правила) двух различных выводов, приводящих исходные деревья к представлению, имеющему некоторую заранее заданную общность признаков (в частности, одинаковую ЛСК).

Рассмотрим требования, которым должна удовлетворять модель информационного пространства правил Δ-грамматики в целях адекватности рассматриваемой задаче распознавания семантических повторов.

Во-первых, модель должна описывать взаимосвязи между входными и выходными деревьями различных правил.

Во-вторых, модель должна по заданному дереву, функционально соответствующему входу некоторого правила $rule_j \in \Pi$, указать деревья, достижимые из заданного применением последовательности правил с максимальной длиной, равной мощности рассматриваемой системы правил (числу правил в системе), и описать последовательность переходов, соответствующих указанным выводам в Δ-грамматике (последовательность применения правил).

В-третьих, модель должна для двух заданных деревьев Tio_1 и Tio_2, функционально соответствующим входам-выходам некоторых правил множества Π, определить возможность приведения к виду с заданной общностью признаков и указать последовательность выполняемых преобразований.

На основе выдвинутых требований построим формальную концептуальную модель системы синонимического перифразирования глубинных синтаксических структур, представляемой в терминах расширенной лексико-синтаксической Δ-грамматики (1.2). Следует особо подчеркнуть, что объектом моделирования здесь являются структурные взаимосвязи между правилами, которые не могут быть напрямую заданы в рамках Δ-грамматик, поскольку последние описывают трансформации в языке с точки зрения отдельных преобразований.

Рассмотрим множество T_{IO} входов и выходов правил $rule_j \in \Pi$, составляющих систему, в качестве множества объектов информационного пространства (множества информационных элементов) заданной Δ-грамматики. При этом T_{IO} есть объединение множества входов T_I и множества выходов T_O, а модель совокупности правил системы есть совокупность сетей Петри, построенных из моделей отдельных правил как примитивов.

В соответствии с описанием (2.8) для сети N_i одной отдельно взятой i-й системы правил множество позиций P_i включает те элементы множества T_{IO},

которые соответствуют входам и выходам правил, способных образовывать систему. Множество возможных переходов T_i сети составляют переходы между состояниями, соответствующими входным и выходным деревьям правил. Исходя из содержательной особенности системы правил Δ-грамматики, число позиций сети N_i, инцидентных переходу, не превышает 1. Это ограничение не является свойством топологии сети, а естественным образом вытекает из ограничений, накладываемых на примитивы. Мощность множества переходов при этом не превышает величины $\frac{|P_i|!}{(|P_i|-2)!}$. Следует также подчеркнуть следующую особенность матриц инцидентности: $\sum_{k=1}^{|P_i|} F_{kj} = 1$ для $\forall j = 1,\ldots,|T_i|$ и $\sum_{j=1}^{|T_i|} H_{kj} = 1$ для $\forall k = 1,\ldots,|P_i|$. В содержательном плане это означает, что одно и то же правило Δ-грамматики не может описывать генерацию двух различных деревьев, в то же время к одному и тому же дереву может быть применено несколько правил заданной системы.

Теорема 2.1. Пусть N_i – сеть Петри, построенная из примитивов, каждый из которых моделирует работу правила из некоторого подмножества правил заданной Δ-грамматики, образующих систему. Тогда сеть N_i является безопасной в течение всего времени функционирования системы.

Доказательство. Действительно, согласно определению, сеть Петри безопасна, если любая ее позиция содержит не более одной фишки. За одно срабатывание правила $rule_j \in \Pi$ не может быть обработано более одного дерева. А это означает, что число маркеров (фишек) в позиции сети (2.8) не превышает 1. С другой стороны, для $\forall t_{ji} \in T_i$ $\sum_{k=1}^{|P_i|} H_{jk} = 1$, что говорит о невозможности появления в любой позиции $p_{ki} \in P_i$ более чем одного маркера и тем самым доказывает теорему.

Активизация (установка начальной разметки M_{0i}) сети N_i соответствует активизации позиции для входа/выхода того правила, которому функционально соответствует входное дерево. Начальная маркировка или разметка, как и любая другая из допустимых в рассматриваемой сети разметок, характеризуется тем, что:

– число маркеров (фишек) в одной позиции не превышает 1, $M_{0i} : P_i \to \{0,1\}$;

– одновременно активизированными могут быть не более одной позиции.

Сеть N_i обладает рядом свойств, касающихся переходов от разметки к разметке.

Во-первых, любая из допустимых для сети разметок может выступать в роли начальной, поскольку изначальная активизация той или иной позиции зависит от того, входу или выходу какого правила системы функционально соответствует входное дерево.

Во-вторых, не исключается наличие тупиковых разметок, обусловленное особенностями моделируемых систем правил. Описываемые Δ-грамматиками реальные системы правил могут включать односторонние преобразования. Для русского языка примером могут служить смысловые импликации [62, с. 158–159].

В-третьих, начальная разметка может оказаться тупиковой. Этому соответствует ситуация, когда входное дерево функционально соответствует выходу одностороннего преобразования.

Последовательность применяемых правил моделируется последовательностью $\tau = (t_{1i}, t_{2i}, \ldots, t_{ki})$ срабатываний переходов:

$$Tio_1 \xrightarrow{rule_1(r_{12})} Tio_2 \xrightarrow{rule_2(r_{23})} Tio_3 \to \ldots \to Tio_k \xrightarrow{rule_k(r_{k,k+1})} Tio_{k+1}, \qquad (2.9)$$

где $t_{1i} \Leftrightarrow rule_1(r_{12})$, $t_{2i} \Leftrightarrow rule_2(r_{23})$, … , $t_{ki} \Leftrightarrow rule_k(r_{k,k+1})$. При этом происходит последовательная смена разметок:

$$M_{0i} \xrightarrow{t_{1i}} M_{1i} \xrightarrow{t_{2i}} M_{2i} \to \ldots \to M_{k-1,i} \xrightarrow{t_{ki}} M_{ki}, \qquad (2.10)$$

где $M_{0i} \Leftrightarrow Tio_1$, $M_{1i} \Leftrightarrow Tio_2$, … , $M_{k-1,i} \Leftrightarrow Tio_k$, $M_{ki} \Leftrightarrow Tio_{k+1}$.

При этом множество разметок, достижимых из начальной разметки M_{0i} и образующих множество достижимости $R(N_i)$ сети N_i находится в зависимости от задания M_{0i}. Если входное дерево функционально соответствует выходу одностороннего преобразования в рассматриваемой Δ-грамматике (тупиковая разметка), то $R(N_i) = \{M_{0i}\}$. Максимальная мощность множества $R(N_i)$ для системы из n_Γ правил будет иметь место тогда, когда начальная разметка M_{0i} соответствует активизации позиции $p_{ki} \in P_i$ с минимальным значением суммы $\sum_{j=1}^{n_\Gamma} H_{kj}$. Содержательно такая ситуация означает активизацию входа правила системы, в которой все правила двусторонни, либо активизацию входа одностороннего преобразования, который не является выходом никакого другого правила в рассматриваемой системе.

В соответствии с показанным свойством достижимости разметок, множество $L(N_i)$ символьных цепочек, описывающих последовательности срабатывания переходов и составляющих свободный язык рассматриваемой сети Петри, будет определяться в зависимости от задания начальной разметки M_{0i}. Функционирование системы правил Δ-грамматики описывается указанными символьными цепочками. При этом последовательность $t_{1i}, t_{2i}, \ldots, t_{k-1,i}, t_{ki}$ срабатывания переходов есть слово τ в языке $L(N_i)$.

Задача приведения деревьев Tio_1 и Tio_{k+1} к виду с одинаковой ЛСК фактически включает в себя три задачи, связанные с исследованием свойств сети, построенной из моделей правил как примитивов:

1) определение достижимости разметки M_{ki} из начальной разметки M_{0i}. Данная задача есть поиск слова $\tau \in T_i^* \,|\, M_{0i} \xrightarrow{\tau} M_{ki}$, где T_i^* – множество всех слов в алфавите T_i;

2) задача обратимости слова τ: если $\tau \in T_i^* \left| M_{0i} \xrightarrow{\tau} M_{ki} \right.$, то существует ли слово $\tau' = \left(t'_{ki}, t'_{k-1,i}, \ldots, t'_{2i}, t'_{1i}\right)$:

$$M_{0i} \xleftarrow{t'_{1i}} M_{1i} \xleftarrow{t'_{2i}} M_{2i} \leftarrow \ldots \leftarrow M_{k-1,i} \xleftarrow{t'_{ki}} M_{ki}, \qquad (2.11)$$

где $M_{0i} \Leftrightarrow Tio_1$, $M_{1i} \Leftrightarrow Tio_2$, … , $M_{ki} \Leftrightarrow Tio_{k+1}$;

3) задача определения оптимального слова $\tau \in T_i^* \left| M_{0i} \xrightarrow{\tau} M_{ki} \right.$. Суть: если существуют $\tau_1, \tau_2, \ldots, \tau_l$: $M_{0i} \xrightarrow{\tau_1} M_{ki}$, $M_{0i} \xrightarrow{\tau_2} M_{ki}$, …, $M_{0i} \xrightarrow{\tau_l} M_{ki}$, то в качестве оптимального берется слово наименьшей длины, причём предпочтение всегда отдаётся обратимому слову.

Отметим, что в отличие от первых двух, третья задача относится к задачам анализа динамики функционирования системы. Действительно, здесь для заданной Δ-грамматики требуется исследовать возможные последовательности срабатываний правил. Решение подобных задач, согласно классической теории сетей Петри, определяется тем, к какому классу языков [53] относится язык $L\left(N_i\right)$, порождаемый заданной сетью N_i.

Проведем предварительный анализ языка рассматриваемой сети Петри для отнесения к одному из классов, представленных в [53]. Рассмотрим системные события, соответствующие переходам сети, с точки зрения их тождественности, позволяющей рассматривать одни переходы как “одинаковые”, а другие – как “разные”. Можно показать, что в сети Петри, построенной из моделей отдельных правил Δ-грамматики, все переходы будут различны. Сформулируем данное утверждение в виде леммы и теоремы.

Лемма 2.1. Пусть Γ – расширенная лексико-синтаксическая Δ-грамматика вида (1.2). Все правила $rule_j \in \Pi$ указанной грамматики, относящиеся к произвольным элементарным преобразованиям, являются взаимно различными.

Доказательство. Пусть $\Pi_U \subset \Pi$ – множество универсальных специальных преобразований типа расщепления, перевешивания узла и склеивания узлов де-

рева. Согласно доказанной в [14] для синтаксических Δ-грамматик теореме о моделировании произвольного элементарного преобразования специальными, произвольное элементарное преобразование $rule_j \in (\Pi \setminus \Pi_U)$ моделируется конечной последовательностью $rule_{1j}, rule_{2j}, \ldots, rule_{mj}$ правил $rule_{kj} \in \Pi_U$. Эта же теорема означает, что правила $rule_{kj} \in \Pi_U$ выполняются группами, внутри которых существует жесткий порядок, и каждая группа соответствует одному правилу $rule_j \in (\Pi \setminus \Pi_U)$. Таким образом, существует взаимно-однозначное соответствие между элементами множества произвольных элементарных преобразований $\Pi \setminus \Pi_U$ и конечными последовательностями правил $rule_{kj} \in \Pi_U$. Утверждение о наличии двух одинаковых последовательностей правил из множества Π_U, моделирующих два произвольных элементарных преобразования $rule_1 \in (\Pi \setminus \Pi_U)$ и $rule_2 \in (\Pi \setminus \Pi_U)$, $rule_1 \neq rule_2$, противоречит условию указанной теоремы. Кроме того, наличие двух правил $\{rule_1, rule_2\} \subset (\Pi \setminus \Pi_U)$, моделируемых одной и той же последовательностью универсальных специальных преобразований $rule_{1j}, rule_{2j}, \ldots, rule_{mj}$, означало бы отсутствие ограничений на число произвольных элементарных преобразований, эквивалентных заданному $rule_j \in (\Pi \setminus \Pi_U)$, что противоречит условию конечности множества элементарных преобразований. Таким образом, среди правил множества $\Pi \setminus \Pi_U$ нельзя выделить пары одинаковых, что и требовалось доказать.

Теорема 2.2. Все символы-переходы $t_{ji} \in T_i$ сети N_i различны.

Доказательство. Согласно определению сети N_i, каждый символ-переход t_{ji} соответствует некоторому произвольному элементарному преобразованию $rule_j \in (\Pi \setminus \Pi_U)$, выполняемому в одну сторону; двустороннему преобразованию соответствуют два различных символа-перехода t_{ji} и t'_{ji}, $t_{ji} \neq t'_{ji}$. Как следует из доказанной *леммы 2.1*, среди преобразований из множества $\Pi \setminus \Pi_U$ нет пары

одинаковых. Следовательно, нет одинаковых и среди символов-переходов $t_{ji} \in T_i$, что и требовалось доказать.

Из доказанной теоремы следует, что помечающая функция $\sum : T_i \to Alph$ для сети N_i сопоставляет каждому переходу $t_{ji} \in T_i$ единственный символ алфавита $Alph$, соответствующий обозначению некоторого правила из произвольных элементарных преобразований в заданной Δ-грамматике.

Будучи помеченной, сеть N_i обладает рядом свойств, актуальных для задач достижимости заданной разметки и принадлежности произвольной последовательности символов алфавита $Alph$ языку рассматриваемой сети Петри.

Свойство 1. Некоторая фиксированная разметка M_{fi}, называемая терминальной, допустима в сети N_i тогда и только тогда, когда среди множества правил моделируемой системы имеются односторонние преобразования, выходам которых соответствуют тупиковые разметки.

Свойство 2. Свободный терминальный язык $L(N_i, M_{fi})$ сети N_i, описываемый последовательностями переходов от начальной разметки M_{0i} к фиксированной терминальной разметке M_{fi}, определяется в зависимости от задания M_{0i}.

Свойство 3. Произвольность задания начальной разметки M_{0i} влечет возможность существования нескольких свободных терминальных языков $L(N_i, M_{fi})$ на сети N_i.

Свойство 4. Префиксный язык $\{\sum(\tau) | \tau \in L(N_i)\}$ помеченной сети $(N_i, \sum)$ получается из свободного языка $L(N_i)$ прямой заменой символов-переходов $t_{ji} \in T_i$ на соответствующие символы из $Alph$.

Свойство 5. В сети N_i, помеченной символами алфавита $Alph$, появление λ-переходов (переходов, которым не сопоставляется ни один символ из $Alph$, [53, с.

36]) возможно в случае моделирования некоторого произвольного элементарного преобразования последовательностью универсальных специальных элементарных преобразований. В этом случае соответствующий переход $t_{ji} \in T_i$ замещается последовательностью λ-переходов, соответствующих выполняемым универсальным элементарным преобразованиям, а префиксный язык $\{\sum(\tau) | \tau \in L(N_i)\}$ помеченной сети $(N_i, \sum)$ будет относиться к классу ℓ^{λ} префиксных языков сетей Петри. Верхний индекс λ означает, что помечающая функция может быть частичной, то есть помеченная сеть $(N_i, \sum)$ может содержать λ-переходы. При отсутствии λ-переходов ее префиксный язык будет относиться к подклассу ℓ класса ℓ^{λ} префиксных языков сетей Петри.

Определение достижимости заданной разметки M_{ki} из начальной M_{0i} в сети N_i возможно организацией полного перебора на дереве достижимости с запоминанием последовательности срабатывания переходов и узлов, соответствующих разметкам (состояниям при представлении дерева достижимости как графа состояний). Для целевой разметки M_{ki} поиск слова $\tau \in T_i^* | M_{0i} \xrightarrow{\tau} M_{ki}$ есть поиск пути, описываемого формулой (2.10), по дереву достижимости. Данный путь начинается в M_{0i} и заканчивается в M_{ki}. Следует отметить, что решение задачи определения достижимости заданной разметки идет с запоминанием символов языка $L(N_i)$, составляющих рассматриваемое слово $\tau \in T_i^*$, и последовательности разметок от M_{0i} до M_{ki}, определяющих путь по дереву достижимости. Сказанное необходимо для определения обратимости слова τ.

Замечание. В дереве достижимости сети N_i может быть несколько узлов, соответствующих эквивалентным друг другу разметкам. Во избежание зацикливания алгоритма перебора следует ввести процедуру “отсечения” ветвей дерева достижимости, соответствующих найденному решению, что позволит избежать многократного прохождения по одному и тому же пути при прямом переборе.

Доказанная для ограниченных сетей Петри конечность дерева достижимости [144, 147, 156] позволяет говорить о конечности рассматриваемого процесса перебора. Сформулируем данное свойство сети N_i в виде леммы.

Лемма 2.2. Проблема достижимости заданной разметки M_{ki} из начальной M_{0i} в сети N_i разрешима.

Замечание 1. Поскольку число позиций в сети N_i равно числу правил рассматриваемой *i*-й системы правил n_np_i и число фишек в каждой из позиций не превышает 1, то число узлов дерева достижимости не превышает $2^{n\_np_i+1}-1$.

Замечание 2. Задача достижимости перехода в сети N_i принадлежит к классу NP-трудных задач. Истинность этого утверждения очевидна, поскольку, как показано в [144], известная NP-трудная задача выполнимости конъюнктивной нормальной формы сводится к задаче достижимости перехода в ограниченной сети Петри.

Рассмотрим более подробно задачу определения оптимального слова $\tau \in T_i^* \left| M_{0i} \xrightarrow{\tau} M_{ki} \right.$ для языка $L(N_i)$ рассматриваемой сети N_i.

Как было показано ранее, задача поиска оптимального слова τ, описывающего последовательность смены состояний системы между некоторой разметкой M_{0i}, выбранной в качестве начальной, и заданной разметкой M_{ki} включает следующие подзадачи:

– определение достижимости заданной разметки M_{ki} в сети N_i при задаваемой начальной M_{0i};

– в случае нескольких слов $\tau \in T_i^* \left| M_{0i} \xrightarrow{\tau} M_{ki} \right.$ исследование возможностей их обратимости согласно формуле (2.11);

– определение слова минимальной длины среди обратимых слов;

– в случае отсутствия обратимых слов среди $\tau \in T_i^* \left| M_{0i} \xrightarrow{\tau} M_{ki} \right.$ определение среди найденных слов слова минимальной длины.

Рассмотрим в отдельности каждую из упомянутых подзадач.

Теорема 2.3. Проблема определения обратимости слова $\tau \in T_i^* \left| M_{0i} \xrightarrow{\tau} M_i^k \right.$ языка $L(N_i)$ разрешима.

Доказательство. Для любой разметки M_{ki} в сети N_i проблема ее достижимости из выбранной начальной M_{0i} является разрешимой по *лемме 2.2*. Как следует из формулировки задачи нахождения слова $\tau' = (t'_{ki}, t'_{k-1,i}, \ldots, t'_{2i}, t'_{1i})$, описывающего обратную последовательность переходов от M_{ki} к M_{0i}, отличие ее от классической задачи принадлежности некоторого слова языку сети Петри состоит только в том, что неизвестной является последовательность переходов $t'_{ki}, t'_{k-1,i}, \ldots, t'_{2i}, t'_{1i}$ при известной последовательности смены разметок согласно формуле (2.11). При наличии (согласно *лемме 2.1*) взаимно-однозначного соответствия между указанной последовательностью и последовательностью переходов $\tau' = (t'_{ki}, t'_{k-1,i}, \ldots, t'_{2i}, t'_{1i})$ задача сводится к определению принадлежности слова τ' языку $L(N_i)$. Как было показано выше (см. *свойство 5*), префиксный язык $\{\sum(\tau) | \tau \in L(N_i)\}$ помеченной сети (N_i, Σ) относится либо к классу ℓ^λ, либо к подклассу ℓ этого класса языков. Как было доказано в [53, с. 48], проблема принадлежности разрешима для языков класса ℓ и ℓ^λ, что позволяет говорить о разрешимости задачи поиска обратного слова и для слова $\tau \in T_i^* \left| M_{0i} \xrightarrow{\tau} M_{ki} \right.$ в языке $L(N_i)$. Теорема доказана.

Определение слова наименьшей длины среди обратимых $\tau \in T_i^* \left| M_{0i} \xrightarrow{\tau} M_{ki} \right.$ делается разрешимым введением функции $f(M_{ki})$ оценки стоимости пути по дереву достижимости (стоимость пути определяется числом срабатываемых переходов) от заданной начальной разметки к целевой. При этом при просмотре дерева достижимости от каждой последующей вершины к вершине, породившей её, проводятся указатели, позволяющие восстановить путь назад

к корню дерева достижимости после обнаружения целевого узла. Для каждого целевого узла дерева достижимости вычисление функции $f(M_{ki})$ и запоминание ее значения производится при просмотре указателей в обратном направлении – от цели к началу. Во избежание зацикливания поискового алгоритма здесь необходимо ввести процедуру отсечения ветвей дерева достижимости, соответствующих найденному решению.

Таким образом, все задачи, к которым сводится поиск оптимального слова $\tau \in T_i^* | M_{0i} \xrightarrow{\tau} M_{ki}$, разрешимы, следовательно, проблема поиска оптимального слова языка $L(N_i)$ разрешима. Сформулируем данное утверждение в виде теоремы.

Теорема 2.4. Пусть N_i - сеть Петри, построенная из примитивов вида (2.8). Тогда проблема определения оптимального слова $\tau \in T_i^* | M_{0i} \xrightarrow{\tau} M_{ki}$ (T_i^* – множество всех слов в алфавите T_i) в языке $L(N_i)$ является разрешимой.

С целью сокращения перебора при определении оптимального слова $\tau \in T_i^* | M_{0i} \xrightarrow{\tau} M_{ki}$, будем рассматривать состояние системы, моделируемой сетью N_i, одновременной активизацией не одного, а двух информационных элементов, соответствующих сценарию согласно данному в [147] определению.

Действительно, рассмотренная сеть N_i позволяет отражать активизацию только одного информационного элемента, в то время как состояние системы в решаемой задаче построения совокупности целевых выводов в Δ-грамматике описывается как минимум двумя одновременно активизированными элементами, соответствующими входам-выходам различных правил. Более того, в составе определенной таким образом динамической информационной модели отсутствует важный компонент, позволяющий задать реальное целевое состояние системы, которое отличается от описываемого в модели посредством разметки M_{ki} в случае, если оптимальное слово $\tau \in T_i^* | M_{0i} \xrightarrow{\tau} M_{ki}$ является обратимым.

Иными словами, если в системе правил существует минимальная последовательность двусторонних преобразований $rule_1, rule_2, \ldots, rule_{k-1}, rule_k$, где входам/выходам правил $rule_1$ и $rule_k$ функционально соответствуют исходные деревья, то целевое состояние, соответствующее одинаковой ЛСК, должно быть равноудалено и от входа $rule_1$, и от входа $rule_k$. При найденном обратимом оптимальном слове $\tau \in T_i^* \left| M_{0i} \xrightarrow{\tau} M_{ki} \right.$ это означает описать две последовательности переходов τ_1 и τ_1' таких, что

$$\tau_1 = (t_{1i}, t_{2i}, \ldots, t_{li}) \colon \tau_1 \subset \tau,\ \tau_1 \in T_i^* \left| M_{0i} \xrightarrow{\tau_1} M_{li} \right., \text{ а}$$

$$\tau_1' = (t_{ki}', t_{k-1,i}', \ldots, t_{li}') \colon \tau_1' \subset \tau',\ \tau_1' \in T_i^* \left| M_{ki} \xrightarrow{\tau_1'} M_{li} \right..$$

Последовательности τ_1 и τ_1' должны удовлетворять следующему требованию: если $|\tau| \bmod 2 = 1$ (длина обратимого оптимального слова нечётна), то $|\tau_1| = |\tau| \ div\ 2 + 1$ и $|\tau_1'| = |\tau| \ div\ 2$, а если $|\tau| \bmod 2 = 0$ (длина обратимого оптимального слова чётна), то $|\tau_1| = |\tau_1'| = |\tau| \ div\ 2$. Иными словами, на основе найденного обратимого оптимального слова $\tau \in T_i^* \left| M_{0i} \xrightarrow{\tau} M_{ki} \right.$, однозначно определяющего достижимость разметки M_{ki} из заданной начальной M_{0i} в сети N_i, требуется найти последовательности переходов τ_1 и τ_1' к разметке M_{li}, равноудаленной от разметок M_{0i} и M_{ki}.

Модель системы правил $rule_j \in \Pi$ переходит из состояния в состояние путем активизации различных пар $\{Tio_1, Tio_2\} \subset T_{IO}$. Сценарии над множеством информационных элементов образуют пары, составляющие подмножество декартова произведения $T_{IO} \times T_{IO}$.

Обозначим множество всех сценариев внутри системы правил, моделируемой сетью N_i, как Sc_i. Формально пара $\{Tio_1, Tio_2\}$ соответствует некоторому

сценарию $Sc_{ji} \in Sc_i$. Целевое состояние характеризуется наличием двух фишек в некоторой позиции $p_{ji} \in P_i$ и ему соответствует сценарий $Sc_{ji} = \{Tio_1, Tio_2\}$, для которого $Tio_1 = Tio_2$.

При срабатывании одного из переходов, допустимых в рамках сценария Sc_{ji}, изменяется активность только одного из двух входящих в сценарий информационных элементов. В некоторый момент времени система правил Δ-грамматики (1.2), моделируемая сетью N_i, может перейти от сценария $Sc_{ji} = \{Tio_1, Tio_2\}$ к сценарию $Sc_{ki} = \{Tio_3, Tio_4\}$, где $j,k \in 1,\ldots,|Sc_i|$, только при наличии в указанной системе пары правил $\{rule_1, rule_2\}$ таких, что:

– либо $rule_1(r_{13}): Tio_1 \xrightarrow{rule_1(r_{13})} Tio_3$, $rule_2(r_{24}): Tio_2 \xrightarrow{rule_2(r_{24})} Tio_4$ и $r_{13} \wedge r_{24} = true$;

– либо $rule_1(r_{14}): Tio_1 \xrightarrow{rule_1(r_{14})} Tio_4$, $rule_2(r_{24}): Tio_2 \xrightarrow{rule_2(r_{24})} Tio_4$ и $r_{14} \wedge r_{23} = true$.

Пусть

$rule_1(r_{13}): Tio_1 \xrightarrow{rule_1(r_{13})} Tio_3 \Leftrightarrow t_{1i}$, $rule_2(r_{24}): Tio_2 \xrightarrow{rule_2(r_{24})} Tio_4 \Leftrightarrow t_{2i}$, $Tio_1 \Leftrightarrow p_{1i}$, $Tio_2 \Leftrightarrow p_{2i}$, $Tio_3 \Leftrightarrow p_{3i}$, $Tio_4 \Leftrightarrow p_{4i}$, $\{p_{1i}, p_{2i}, p_{3i}, p_{4i}\} \subset P_i$, при этом $\{t_{1i}, t_{2i}\} \subset T_i$ для сети N_i. Обозначим $\{Tio_3, Tio_2\}$ как Sc_{1i}, а $\{Tio_1, Tio_4\}$ – как Sc_{2i}. Используя терминологию работ [144, 147], будем говорить, что сценарий Sc_{ji} связывается со сценарием Sc_{1i} через переход t_{1i} (обозначается: $Sc_{ji} \xrightarrow{t_{1i}} Sc_{1i}$), а со сценарием Sc_{2i} – через переход t_{2i} (соответственно, $Sc_{ji} \xrightarrow{t_{2i}} Sc_{2i}$). Аналогично сценарий Sc_{1i} связывается со сценарием Sc_{ki} через переход t_{2i}, а сценарий Sc_{2i} со сценариаем Sc_{ki} – через переход t_{1i}.

Теорема 2.5. Для каждого задаваемого над сетью N_i сценария $Sc_{ji} \in Sc_i$ можно указать максимум два различных перехода $t_{1i} \in T_i$ и $t_{2i} \in T_i$ таких, что существуют взаимно различные сценарии Sc_{1i} и Sc_{2i} из множества Sc_i и при этом $Sc_{ji} \xrightarrow{t_{1i}} Sc_{1i}$, $Sc_{1i} \xrightarrow{t_{2i}} Sc_{ki}$, $Sc_{ji} \xrightarrow{t_{2i}} Sc_{2i}$, $Sc_{2i} \xrightarrow{t_{1i}} Sc_{ki}$.

Доказательство. Срабатывание одного перехода $t_{ji} \in T_i$ ведет к изменению активности максимум одной позиции $p_{ki} \in P_i$, поскольку для $\forall j = 1,\ldots,|T_i|$ $\sum_{k=1}^{|P_i|} H_{jk} = 1$. Согласно *теореме 2.2* в сети N_i все символы-переходы различны. Следовательно, два различных сценария над рассматриваемой сетью могут быть связаны только одним переходом. А поскольку число активных позиций сети N_i в рамках сценария равно двум, то существует максимум два пути:

$$Sc_{ji} \xrightarrow{t_{1i}} Sc_{1i} \xrightarrow{t_{2i}} Sc_{ki} \text{ и } Sc_{ji} \xrightarrow{t_{2i}} Sc_{2i} \xrightarrow{t_{1i}} Sc_{ki},$$

где $Sc_{1i} \neq Sc_{2i}$, $t_{1i} \neq t_{2i}$, $Sc_{ji} \cap Sc_{1i} \neq \varnothing$, $Sc_{1i} \cap Sc_{ki} \neq \varnothing$, $Sc_{ji} \cap Sc_{2i} \neq \varnothing$, $Sc_{2i} \cap Sc_{ki} \neq \varnothing$ (рис. 2.1), что и служит доказательством теоремы.

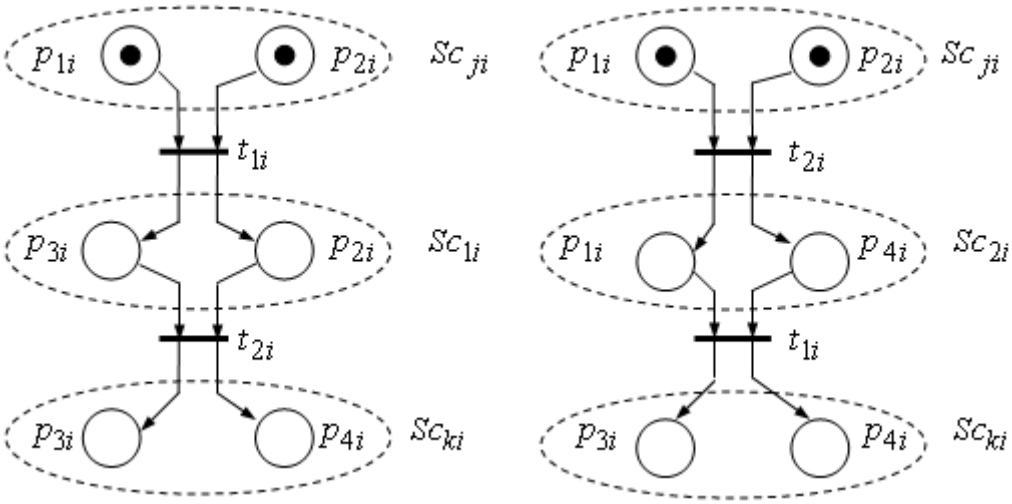


Рис. 2.1. Две возможные последовательности переходов от сценария Sc_{ji} к сценарию Sc_{ki}

Заметим, что при достижении целевого состояния системы (решение найдено) информационный элемент, соответствующий целевому состоянию, будет

активизирован дважды. Поэтому максимальное число сценариев, задаваемых над множеством позиций P_i, равно числу комбинаций из $|P_i|$ по 2 (случаи активизации различных информационных элементов) плюс мощность множества P_i:

$$\max|Sc_i| = \frac{|P_i|!}{2(|P_i|-2)!} + |P_i|.$$

Рассмотрим символьное описание сети N_i, допускающее ее машинную обработку и хранение, с использованием представления сценария Sc_{ji} как совокупности двух одновременно активизированных информационных элементов: $Sc_{ji} = \{p_{li}, p_{mi}\} \subset P_i$, где $l, m \in 1,\ldots,|P_i|$, а $j \in 1,\ldots,|Sc_i|$. Поскольку в рамках одного и того же сценария может быть разрешено несколько переходов, то его описание будет выглядеть следующим образом:

$$Sc_{ji} = \{Sc_{ki},\ldots,Sc_{ni}, p_{li}, p_{mi}\}, \quad (2.12)$$

где $Sc_{ki},\ldots,Sc_{ni}$ – множество сценариев, связанных с Sc_{ji} через некоторые переходы из множества T_i, $k,n \in 1,\ldots,|Sc_i|$.

Множество сценариев, связанных со сценарием Sc_{ji} через разрешенные в его рамках переходы из множества T_i, можно представить массивом ссылок ref_{ji}. Обозначив $\{p_{li}, p_{mi}\}$ как P_{ji}, преобразуем формулу (2.12):

$$Sc_{ji} = \{ref_{ji}, P_{ji}\}. \quad (2.13)$$

Массив ref_{ji} формируется в зависимости от содержимого P_{ji} посредством обработки матрицы инцидентности F_i. Исходными данными при этом будут:

$\sum_{Ri}$ – массив информации о переходах. Элементами указанного массива являются ссылки на описания условий применимости правил Δ-грамматики, соответствующих переходам $t_{ji} \in T_i$. Каждое из условий определяется выражениями (2.6) и (2.7);

$\sum_{dbfi}$ – массив ссылок на описание входов/выходов правил системы.

Задавая сеть N_i парой массивов $\sum = \left(\sum_{Ri}, \sum_{dbfi}\right)$, можно описать динамику функционирования системы правил Δ-грамматики построением *TS*-сети (ограниченной сети Петри, порождаемой множеством символов-переходов T_i на множестве сценариев Sc_i) на основе задаваемой начальной разметки. При этом указанная разметка соответствует активизации пары позиций сети N_i для входов правил, которым функционально соответствуют исходные деревья. В следующем разделе мы рассмотрим взаимосвязь внутренней структуры входов/выходов правил Δ-грамматики как объектов информационного пространства с информационным наполнением деревьев глубинного синтаксиса.

2.3. Моделирование построения образа суммарного смысла

Предложенная в предыдущем разделе модель учитывает недетерминированный характер порождения Δ-грамматикой множества деревьев. При этом построение целевого вывода сводится к классическим задачам теории сетей Петри. Однако рассмотрение входа/выхода правила в качестве объекта информационного пространства требует формального описания его активизации в зависимости от ситуации использования и с учетом его внутренней структуры. Сказанное предполагает решение двух основных задач:

– построение модели входа/выхода правила как объекта информационного пространства;

– разработка структуры информационного наполнения анализируемого дерева.

При этом основным требованием к модели входа/выхода правила $rule \in \Pi$ в Δ-грамматике (1.2) является отображение различных способов использования при единообразии функционального описания. Анализ вызывающих активиза-

цию входа/выхода правила событий позволяет выделить следующие способы его использования как информационного элемента:

– анализ применимости правила к помеченному дереву с выдачей FALSE/TRUE в качестве результата;

– синтез дерева по задаваемому выходным деревом шаблону;

– распознавание ключевого слова, заменяемого лексическим правилом поддерева;

– расстановка композиционных меток в анализируемом дереве с целью обозначения заменяемого поддерева.

Во всех четырех показанных ситуациях элементы информационного пространства активизируются по-разному ввиду неоднородности вызывающих их активизацию событий при идентичности функциональной структуры процессов активизации. Поскольку задача применения правила к некоторому заданному дереву есть частный случай задачи “Изоморфизм подграфу” [151], то логико-функциональная структура информационного наполнения входного/выходного дерева правила должна быть идентична логико-функциональной структуре информационного наполнения анализируемых деревьев. Говоря об изоморфизме поддереву, будем подразумевать изоморфизм с точностью до функционального соответствия. Само функциональное соответствие определим следующим образом.

Определение 2.7. Деревья Tr_1 и Tr_1' считаются изоморфными с точностью до функционального соответствия, если в дереве Tr_1' из узла α_{11}' в узел α_{12}' идет ветвь с некоторой пометкой тогда и только тогда, когда в дереве Tr_1 из узла α_{11} в узел α_{12} идет ветвь с той же пометкой. При этом узел α_{11}' должен отвечать требованиям, содержащимся в узле α_{11}, а узел α_{12}', соответственно, требованиям, содержащимся в узле α_{12}. В таком случае считается, что узел α_{11}' функционально соответствует узлу α_{11}, а узел α_{12}' – узлу α_{12}.

Рассмотрим структуру информационного наполнения узла дерева на входе/выходе правила, унифицируемую со структурой соответствующего описания для анализируемых деревьев и ориентированную на представление динамических структур данных средствами декларативных языков.

В соответствии с приведенным в работах И.А. Мельчука описанием уровня глубинного синтаксиса, в информационном наполнении узла глубинной синтаксической структуры следует выделить:

– лексическую часть, соответствующую представленному в узле элементу множества W_R модели (1.2);

– грамматическую часть, содержащую семантические словоизменительные характеристики.

Кроме того, в описание узла должны быть введены особые элементы, соответствующие пометке входящей в узел ветви и композиционной метке.

Представим дерево глубинного синтаксиса фразы χ упорядоченной двойкой

$$Tr_{\chi} = \langle W_{\chi}, V_{\chi} \rangle, \tag{2.15}$$

где W_{χ} есть множество узлов, а V_{χ} есть множество ветвей дерева. Информационное наполнение отдельного узла $w_{\chi i} \in W_{\chi}$ может быть представлено списком из четырех элементов:

$$w_{\chi i} = (lx_{\chi i}, gr_{xi}, ar_{\chi i}, cl_{\chi i}). \tag{2.16}$$

Здесь элемент $lx_{\chi i}$ соответствует лексической, $gr_{\chi i}$ – грамматической части узла, $ar_{\chi i}$ – пометке входящей ветви, а $cl_{\chi i}$ – композиционной метке узла. Следует отметить, что $cl_{\chi i}$ является необязательным (факультативным) элементом в списке (2.16) и вводится для обозначения того факта, что рассматриваемый узел является выделенным и участвует в некотором преобразовании исходного дерева.

Как показано в [14], дерево Tr_2 получается из дерева Tr_1 применением элементарного преобразования $t_1 \Rightarrow t_2 | f$ при задаваемом функцией f однозначном отображении множества узлов дерева t_1 во множество узлов дерева t_2, если Tr_1 и Tr_2 представимы, соответственно, в виде:

$$Tr_1 = Com\left(Tr_{01}; \alpha_0 \middle| Com\left(t_1; \alpha_1, \alpha_2, \ldots, \alpha_n \middle| Tr_{11}, Tr_{21}, \ldots, Tr_{n1}\right)\right), \quad (2.17)$$

$$Tr_2 = Com\left(Tr_{01}; \alpha_0 \middle| Com\left(t_2; f(\alpha_1), f(\alpha_2), \ldots, f(\alpha_n) \middle| Tr_{11}, Tr_{21}, \ldots, Tr_{n1}\right)\right), \quad (2.18)$$

Com – операция композиции. Она определяется следующим образом. Пусть в дереве Tr_{01} выделен узел α_0, а в дереве t_1 выделено n узлов $\alpha_1, \alpha_2, \ldots, \alpha_n$ (не обязательно попарно различных). Тогда дерево Tr_1 получается из Tr_{01} в два этапа: “наклеивание” вершин деревьев $Tr_{11}, Tr_{21}, \ldots, Tr_{n1}$ на узлы $\alpha_1, \alpha_2, \ldots, \alpha_n$ дерева t_1 и последующее “наклеивание” вершины получившегося дерева на узел α_0 дерева Tr_{01}. Будем в дальнейшем называть дерево Tr_{01} деревом верхнего контекста (верхним древесным контекстом) заменяемого правилом дерева t_1, а деревья $Tr_{11}, Tr_{21}, \ldots, Tr_{n1}$ – деревьями, соответственно, нижнего контекста (нижним древесным контекстом) заменяемого дерева.

В том случае, когда узел $w_{\chi i} \in W_\chi$ является выделенным, композиционная метка $cl_{\chi i}$ присутствует в списке (2.16) и принимает значения:

– равное 0 для дерева Tr_{01} и обозначает место “крепления” заменяемого (t_1) и заменяющего (t_2) деревьев к Tr_{01};

– в диапазоне от 1 до n – для деревьев нижнего контекста. Каждая из меток $1, \ldots, n$ обозначает место крепления соответствующего дерева $Tr_{11}, Tr_{21}, \ldots, Tr_{n1}$ к заменяемому (заменяющему) дереву.

Лексическая часть $lx_{\chi i}$ узла $w_{\chi i} \in W_\chi$ представляется списком вида:

$$lx_{\chi i} = \left(C_0, fun_n, \ldots, fun_1\right),$$

где C_0 представляет некоторую самостоятельную лексему, лексической производной от которой (в виде последовательно взятых значений лексических функций из списка $fun_n,\ldots, fun_1$) является лексема, соответствующая содержимому узла на поверхностно-синтаксическом уровне. При этом список $fun_n,\ldots, fun_1$ может быть пустым в случае отображения в узле фиктивной лексемы, идиомы либо самостоятельной лексемы, не являющейся лексическим коррелятом других лексем, присутствующих в той же глубинной синтаксической структуре.

Грамматическая часть $gr_{\chi i}$ узла $w_{\chi i} \in W_\chi$ представляется упорядоченной двойкой:

$$gr_{\chi i} = (psp, lstsc),$$

где psp – символьное обозначение части речи (табл. 2.1), $lstsc$ – список семантически обусловленных словоизменительных категорий, обсуждавшихся в [62, с. 144]. У существительных к числу таковых относится число, у глаголов – вид, время, наклонение.

Таблица 2.1

Символьные обозначения частей речи

psp	*Часть речи*	*psp*	*Часть речи*
S	существительное	Conj	союз
V	глагол	Num	числительное
A	прилагательное	P	причастие
Adv	наречие	Prep	предлог

Элемент $ar_{\chi i}$ в составе списка (2.16) принимает целочисленные значения одного из шести типов связей между родительским и дочерним узлом в глубинной синтаксической структуре, а для вершины дерева элемент $ar_{\chi i}$ имеет значение 0 (входящая ветвь отсутствует).

Описание информации узла $w_{\chi i} \in W_{\chi}$ в виде списка (2.16) позволяет:

– формально определить функциональные требования к узлу ГСС при описании компонент дерева, заменяемого некоторым лексическим правилом. При этом символ C_0 выступает в качестве служебного: им задается местонахождение ключевого слова ЛСК;

– при реализации рассматриваемых преобразований деревьев на языке Лисп организовать вычисление значения суперпозиции лексических функций из списка $fun_n, \ldots, fun_1$ с использованием их имен в качестве функциональных аргументов.

Если дерево глубинного синтаксиса фразы χ представить упорядоченной двойкой вида (2.15), то для машинного представления входа/выхода некоторого правила исследуемой Δ-грамматики в целях учета динамики процесса применения этого правила к конкретному дереву целесообразно ввести следующую тройку:

$$Tio_k = \langle Wio_k, Vio_k, Aio_k \rangle, \qquad (2.19)$$

где Wio_k есть множество требований к содержимому узлов, Vio_k – множество требований к разметке ветвей дерева. Компонент Aio_k в терминологии теории графов есть матрица смежности, каждый элемент Aio_{kij} принимает одно из двух возможных значений:

– 1, если в дереве существует ветвь от узла wt_i к узлу wt_j, где $\{wt_i, wt_j\} \subset Wio_k$;

– 0 – в противном случае.

Само дерево при этом может быть представлено рекурсивной структурой данных, каждый элемент которой будет содержать описание вершины четвёркой (2.16) и список дочерних поддеревьев.

В качестве примера на рис. 2.2 приведено списочное описание (в нотации Microsoft muLISP) для входа лексического правила № 17 с обслуживающим его

синтаксическим правилом № 6 в составе системы синонимического перифразирования русского языка[1].

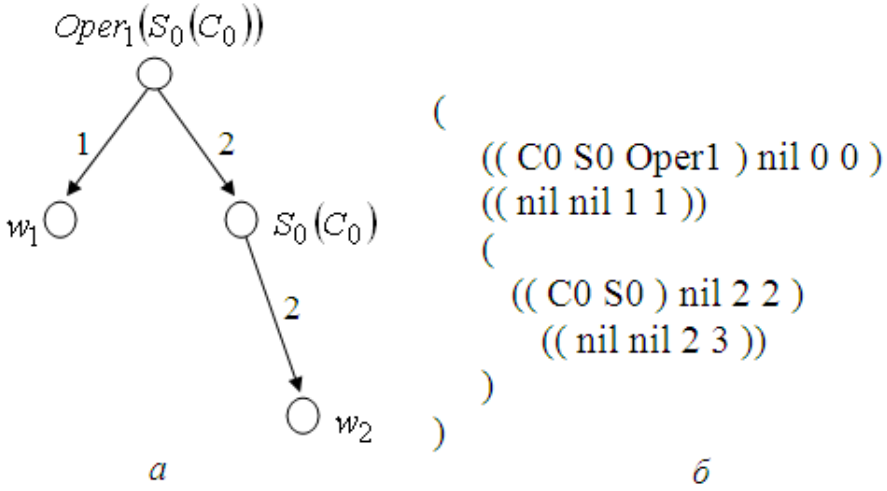


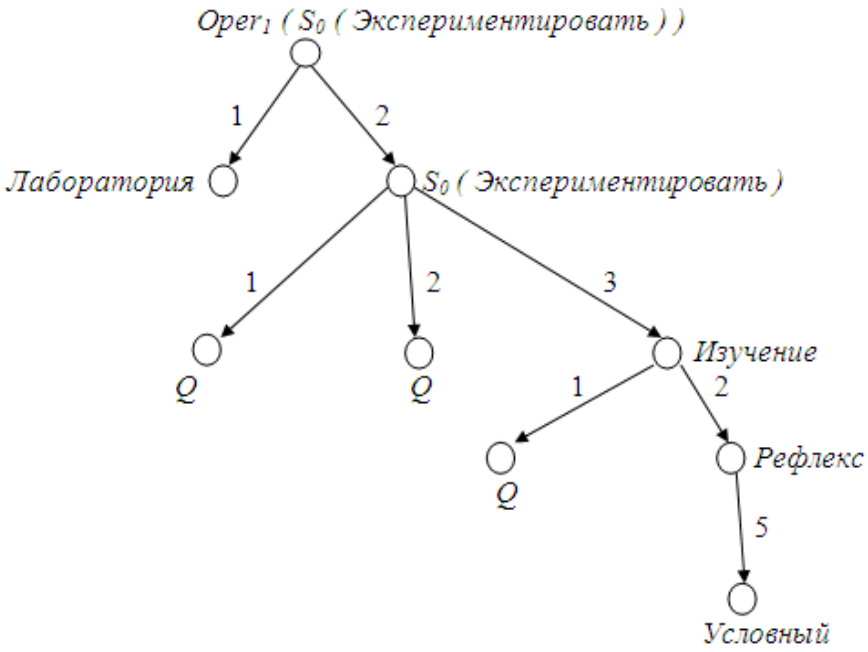
Рис. 2.2. Входное дерево правила Δ-грамматики:
а – графическое представление[2]; *б* – списочное описание в нотации языка Лисп

Как видно из указанного примера, существенной особенностью представления информации узлов входного дерева правила Δ-грамматики является отсутствие определения отдельных компонент дерева. В частности, это относится к требованиям, предъявляемым к лексической и грамматической части узлов входного дерева синтаксическими правилами, обслуживающими лексические замены. В таком случае считается, что соответствующий компонент списка вида (2.16) имеет пустое или неопределенное значение, то есть *nil*.

Действительно, в общем случае лексическое синонимическое преобразование дерева глубинного синтаксиса обслуживается одним или несколькими синтаксическими преобразованиями. Поэтому входное дерево для лексического преобразования следует рассматривать как поддерево входного дерева первого из обслуживающих данную лексическую замену синтаксических преобразований. При этом для синтаксических преобразований значимой является только разметка ветвей, чем и обусловлено присутствие *nil* в качестве значения лексической и грамматической части описания узлов, не входящих в ЛСК. Для сравнения на рис. 2.3

[1] См. [56, с. 154].
[2] Узлы w_1 и w_2 соответствуют произвольным словам, не меняющимся в процессе синонимического перифразирования.

приведено дерево глубинной синтаксической структуры простого распространенного предложения русского языка “*Лаборатория провела эксперименты по изучению условных рефлексов*”.



а

```
(
 (( Экспериментировать S0  Oper1 )
    (V (сов_вид прош_вр изъявит_накл )) 0 nil
 )
   ((( Лаборатория nil )( S (ед_ч)) 1 nil ))
   ((( Экспериментировать S0 )( S (мн_ч)) 2 nil )
    ((( Q nil)( S nil ) 1 nil ))
    ((( Q nil )( S (на)) 2 nil ))
    ((( Изучение nil )( S (по ед_ч)) 3 nil)
      ((( Q nil) nil 1 nil))
      ((( Рефлекс nil)( S (мн_ч)) 2 nil)
        (( Условный nil )(A (мн_ч)) 5 nil)
      )
    )
   )
)
```

б

Рис. 2.3. Анализируемое дерево глубинного синтаксиса:
а – графическое представление; *б* – списочное описание в нотации языка Лисп

Поскольку анализ применимости правил Δ-грамматики для данного предложения не проводился, композиционные метки не определены (в соответствующих местах описания списочного объекта (2.16) на рис. 2.3, *б* стоит *nil*).

При наличии списочного описания для троек вида (2.19) и для двоек вида (2.15), представленного на рис. 2.2, *б* и рис. 2.3, *б*, соответственно, тройка (2.19) может рассматриваться как система, порождающая отличные друг от друга процессы с идентичной функциональной структурой. Прохождение отдельного узла $wt_i \in Wio_k$ при рекурсивной обработке может быть рассмотрено как абстрактное событие, а установление функционального соответствия некоторого узла анализируемого дерева требованиям узла wt_i, размещение в анализируемом узле композиционной метки, синтез дерева по представляемому посредством wt_i шаблону – как разные варианты реализации этого события. Единообразие функционального описания входа/выхода правила $rule \in \Pi$ в (1.2), позволяет рассматривать и анализ применимости этого правила, и синтез дерева, соответствующего выходу правила, как процессы, порождаемые одной и той же сетью Петри:

$$Nt = \{Pt, Trt, Ft, Ht, Ct, Mt_0\}, \qquad (2.20)$$

где множество позиций Pt соответствует множеству состояний информационного элемента, а каждое состояние отождествляется с очередным пройденным узлом $wt_i \in Wio_k$; каждому из переходов $trt_i \in Trt$ соответствует совокупность требований лексической, грамматической части и метки входящей ветви узла wt_i; Ft и Ht есть матрицы инцидентности, аналогичные соответствующим матрицам в (2.8); $Ct = \{c_1, c_2, c_3, c_4\}$ – множество цветов маркера; Mt_0 – начальная разметка. Каждому из цветов маркера соответствует определенный способ использования информационного элемента как вариант разовых реализаций событий прохождения узлов $wt_i \in Wio_k$ при обходе дерева Tio_k: c_1 – анализ применимости правила, c_2 – синтез дерева на выходе правила, c_3 – определение ключевого слова ЛСК, c_4 – расстановка композиционных меток в анализируемом дереве Tr_χ.

Отметим важные особенности сети (2.20), актуальные для моделирования активизации дерева Tio_k как объекта информационного пространства с учетом последовательности действий в процессах, порождаемых входом/выходом правила *rule*. С этой целью проведем предварительный анализ процесса прохождения входного/выходного дерева правила на предмет адекватности процесса, порождаемого сетью Nt. Указанный процесс есть параллельный процесс без альтернатив и конкуренции [53].

Действительно, если представить каждый пройденный узел $wt_i \in Wio_k$ как разовую реализацию факта изменения некоторого условия в системе (в сети Nt указанным изменениям соответствуют элементы множества Pt), а анализ требований лексической, грамматической части и метки входящей ветви узла wt_i – как действие в процессе прохождения дерева Tio_k, то любая пара различных элементов $x \in Pt$ и $y \in Trt$ указанного процесса связаны либо отношениями следствия:

$x\ li\ y \Leftrightarrow (x < y\ or\ y < x) \vee (x = y)$ (здесь *or* есть логическая операция “исключающее или”: $(x < y\ or\ y < x) \Leftrightarrow ((x < y) \wedge \neg(y < x)) \vee (\neg(x < y) \wedge (y < x))$, а запись $x < y$ трактуется таким образом, что изменение условия x завершится до того, как начинается действие y, то есть действие y является следствием изменения условия x),

либо отношением параллелизма: $x\ co\ y \Leftrightarrow (\neg(x < y) \wedge \neg(y < x)) \vee (x = y)$ (здесь запись $x = y$ трактуется таким образом, что действие y и изменение условия x реализуются в процессе независимо друг от друга в том плане, что начало x и завершение y не зависят друг от друга и появление одного элемента в процессе не является следствием появления другого).

При этом не существует пар элементов $\{x, y\}$, связанных отношением альтернативы или конкуренции.

Для разрешения конфликтных ситуаций (когда реализация одного события в системе исключает возможность реализации других, [104, с. 44]) при сетевом

моделировании рекурсивной обработки леса дочерних поддеревьев узла $wt_i \in Wio_k$ топология исходной сети вида (2.20) преобразуется путем замены участка сети, включающего позицию pt_i и инцидентные ей конфликтующие переходы trt_j и trt_l по правилу, показанному на рис. 2.4. Здесь добавляемый переход trt_i' есть безусловный переход, инцидентный позициям pt_i' и pt_i'', каждая из которых представляет собой копию позиции pt_i.

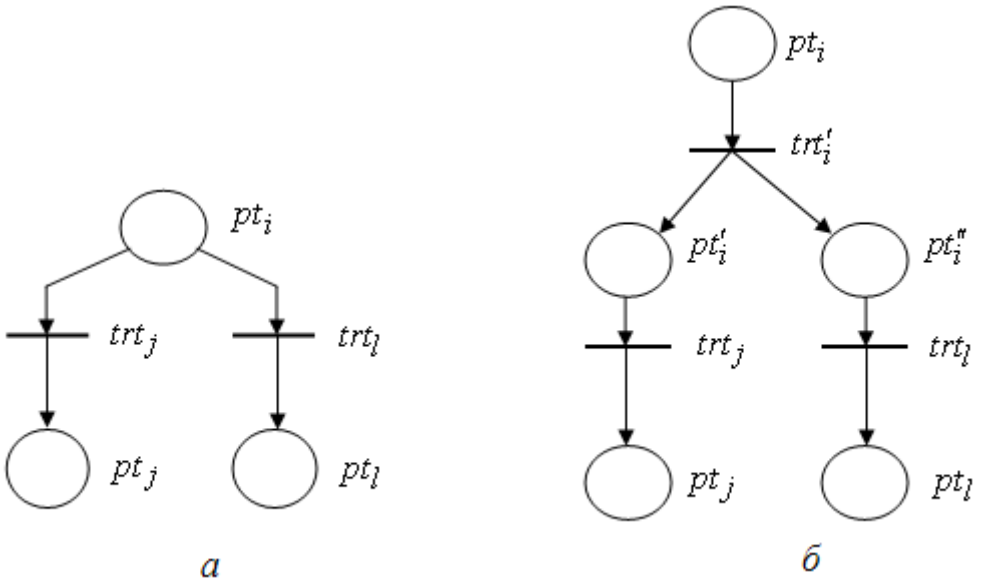


Рис. 2.4. Разрешение конфликта в сети вида (2.20) преобразованием топологии:
а – фрагмент сети до преобразования; *б* – преобразованный фрагмент

Обозначим далее сеть, полученную из исходной сети (2.20) исключением конфликтов по правилу на рис. 2.4, как Nt', $Nt' = \{Pt', Trt', Ft', Ht', Ct, Mt_0'\}$. Для обозначения действия, связанного с окончанием обхода дерева Tio_k, в множество Trt' введён особый переход (обозначим его $tout$), инцидентный тем позициям, которым соответствуют листья в Tio_k. При представлении $Trt' = Trt \cup \{tout\}$ упорядоченным списком будем считать, что переход $tout$ помещается в его конец. Аналогично $Pt' = Pt \cup \{pout\}$, где $pout$ есть позиция, инцидентная единственному переходу $tout$, а при представлении Pt' упорядоченным списком также будем считать, что позиция $pout$ будет находиться в конце этого списка.

В случае успешного анализа применимости правила *rule* к дереву Tr_χ последующая перестройка последнего требует идентификации ключевого слова заменяемой ЛСК и расстановки композиционных меток. Для задания последовательности указанных процессов структура сети Nt' преобразуется введением дополнительной дуги, соединяющей переход *tout* с позицией pt_1, соответствующей началу обхода дерева Tio_k. Преобразованную таким образом сеть Nt' далее обозначим как Nt'',

$$Nt'' = \{Pt', Trt', Ft', Ht'', Ct', Mt_0''\}. \quad (2.21)$$

С целью формализации условия окончания анализа/синтеза (во избежание развертывания бесконечных процессов в сети) множество $Ct' = Ct \cup \{c_5\}$ содержит нейтральный маркер c_5, запрещающий срабатывание перехода, а для перехода *tout* задается индивидуальная таблица условий срабатывания (табл. 2.2).

Таблица 2.2

Условия срабатывания перехода *tout*

$pt_i \in Pt' : Ft'_{ij} = 1$ для $j = \lvert Trt' \rvert$	*pout*	pt_1
c_1	c_3	c_3
c_3	c_4	c_4
c_4	c_5	c_5
c_2	c_5	c_5

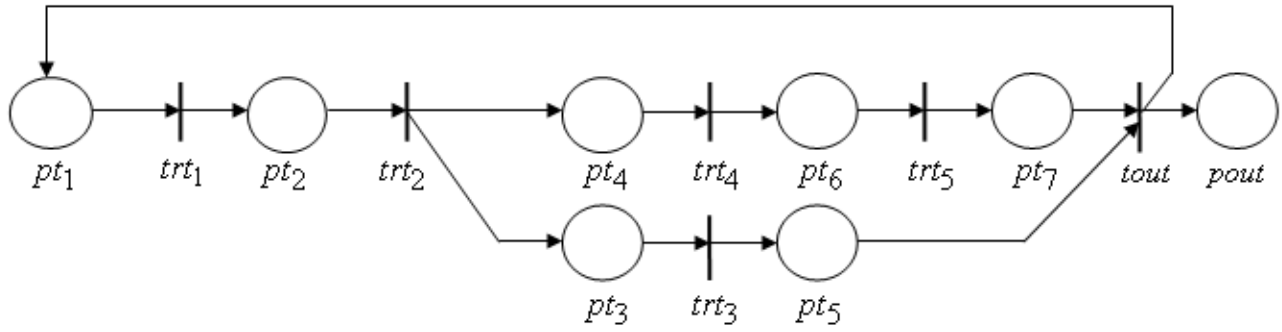


Рис. 2.5. Сетевая модель входа/выхода правила: переход trt_1 соответствует прохождению вершины, trt_3 – узла w_1, trt_4 – узла с содержимым $S_0(C_0)$, trt_5 – узла w_2

На рис. 2.5 показан пример сети Nt'' вида (2.21) для представленного на рис. 2.2, *а* входа правила.

Сеть (2.21) обладает рядом свойств, позволяющих оценить адекватность порождаемых ей процессов моделируемым процессам, порождаемым входом/выходом правила заданной Δ-грамматики как системой при анализе применимости правила к некоторому дереву либо синтезе результирующего дерева по шаблону, определяемому посредством Tio_k .

Лемма 2.3. Сеть Nt'' является конечной.

Доказательство. Сеть $Nt'' = \{Pt', Trt', Ft', Ht'', Ct', Mt_0''\}$, будет конечной при конечности множества позиций Pt' и множества переходов Trt'. Следовательно, чтобы доказать конечность рассматриваемой сети, нужно согласно данному нами её определению доказать конечность сети $Nt' = \{Pt', Trt', Ft', Ht', Ct, Mt_0'\}$. В исходной сети Nt множества позиций Pt и переходов Trt конечны по определению. Следовательно, любые подмножества множеств Pt и Trt будут конечными и число непересекающихся подмножеств на каждом из указанных множеств также будет конечным.

Возьмем некоторую позицию $pt_i \in Pt$: $\sum_{j=1}^{|Trt|} Ft_{ij} > 1$ и построим множество конфликтующих переходов $Tconf \subset Trt$: для $\forall trt_j \in Tconf$ $Ft_{ij} = 1$. Согласно приведенному на рис. 2.4 правилу разрешения конфликта, в множество Pt' будет включено $|Tconf|$ копий позиции pt_i, а множество Trt' пополнится одним безусловным переходом. Множество $Tconf$ есть подмножество Trt, следовательно, оно конечно. Таким образом, множества Pt' и Trt' получаются добавлением в множества Pt и Trt конечного числа элементов, следовательно, сами являются конечными, что и служит доказательством леммы.

Лемма 2.4. Каждая позиция сети $Nt' = \{Pt', Trt', Ft', Ht', Ct, Mt_0'\}$ имеет не более одного входного и не более одного выходного перехода.

Доказательство. Действительно, исходная сеть Nt вида (2.20) обладает тем свойством, что для $\forall j = 1,\ldots,|Pt| \sum_{i=1}^{|Trt|} Ht_{ij} \le 1$, то есть ограничение на число входных переходов для каждой позиции $pt_j \in Pt$ задано исходно. В соответствии с представленным на рис. 2.4 правилом разрешения конфликта сеть Nt преобразуется в Nt' таким образом, что $\sum_{j=1}^{|Trt'|} Ft'_{ij} \le 1$ для $\forall i = 1,\ldots,|Pt'|$, включая вновь добавленные позиции. Рассмотрим, будет ли в Nt' соблюдаться условие $\sum_{i=1}^{|Trt'|} Ht'_{ij} \le 1$. Для вновь добавленных согласно правилу на рис. 2.4 переходов $\sum_{i=1}^{|Pt'|} Ht'_{ji} > 1$, однако для $\forall pt_i \in Pt'$: $Ht'_{ji} = 1$ $\sum_{k=1}^{|Trt'|} Ht'_{ki} = 1$, что и служит доказательством леммы.

Лемма 2.5. Сеть $Nt' = \{Pt', Trt', Ft', Ht', Ct, Mt'_0\}$ не содержит циклов.

Доказательство. При наличии единственного головного места pt_1: $\sum_{j=1}^{|Trt'|} Ht'_{j1} = 0$ и единственного хвостового места $pt_{|Pt'|} \Leftrightarrow pout$: $\sum_{j=1}^{|Trt'|} Ft'_{|Pt'|,j} = 0$ существование цикла в сети Nt' возможно при выполнении одного из двух условий : либо $\exists\, pt_i \in Pt'$: $\sum_{j=1}^{|Trt'|} Ht'_{ji} > 1$, либо $\exists\, trt_i \in Trt'$: $\sum_{j=1}^{|Pt'|} Ft'_{ji} > 1$. Выполнение первого условия невозможно по *лемме 2.4*. Исходная сеть Nt вида (2.20) обладает тем свойством, что для $\forall\, trt_i \in Trt$ $\sum_{j=1}^{|Pt|} Ft_{ji} \le 1$, то есть ограничение на число позиций для каждого перехода задано исходно. Если не принимать во внимание переход $tout \Leftrightarrow trt_{|Trt'|}$, то исходная сеть Nt согласно правилу на рис. 2.4 преобразуется таким образом, что для всех добавляемых переходов $trt_j \in Trt'$ $\sum_{i=1}^{|Pt'|} Ht'_{ji} > 1$, но

$\sum_{i=1}^{|Pt'|} Ft_{ij} = 1$, что ведет к невыполнению второго условия наличия цикла в Nt' и тем самым доказывает лемму.

Теорема 2.6. Все порождаемые сетью Nt'' процессы конечны.

Доказательство. Как следует из *леммы 2.3* и *леммы 2.5*, порождаемый сетью Nt'' процесс будет конечен при соблюдении двух условий: конечности множеств позиций и переходов и отсутствие циклов в сети. Множество позиций Pt' и переходов Trt' сети Nt'', как следует из ее определения, совпадают с соответствующими множествами сети Nt', конечными по *лемме 2.3*. Введение дополнительной дуги, которая соединяет переход $tout \Leftrightarrow trt_{|Trt'|}$ с позицией pt_1: $\sum_{j=1}^{|Trt'|} Ht'_{j1} = 0$ сети Nt', есть зацикливание последней, отличающееся от определенного в [144] зацикливания тем, что головное место pt_1 перехода trt_1 становится хвостовым для $tout$ без добавления нового перехода.

Рассмотрим, приведет ли зацикливание сети Nt' к порождению сетью Nt'' бесконечных процессов. При наложенных таблицей 2.2 ограничений на срабатывание перехода $tout$ при анализе применимости правила с выделением ключевого слова ЛСК и расстановкой композиционных меток в анализируемом дереве Tr_χ (2.15) будет иметь место три последовательных цикла, соответствующих порождаемым сетью Nt'' процессах обхода дерева Tio_k (2.19) непосредственно при анализе применимости правила (маркер c_1), определении ключевого слова ЛСК (маркер c_3) и расстановке композиционных меток в Tr_χ (маркер c_4). Согласно таблице 2.2, появление в позициях, инцидентных переходу $tout$, маркера цвета c_4 означает появление в позициях, которым инцидентен переход $tout$, маркеров цвета c_5, запрещающих срабатывание какого-либо из $trt_i \in Trt'$, что ведет к невозможности каких-либо дальнейших действий в порождаемой здесь сетью Nt'' последовательности процессов, из чего следует ее конечность.

При синтезе дерева по шаблону, задаваемому Tio_k, (маркер цвета c_2), сетью Nt'' порождается единственный процесс обхода этого дерева. Появление, согласно таблице 2.2, в позициях, инцидентных переходу $tout$, маркеров цвета c_2 также означает появление в выходных позициях перехода $tout$ маркеров цвета c_5, что доказывает конечность порождаемого сетью Nt'' процесса обхода дерева (2.19) при синтезе дерева, соответствующего выходу правила Δ-грамматики (1.2). Теорема доказана.

Теорема 2.7. Сеть Nt'' является ограниченной.

Доказательство. Как следует из *теоремы 2.6*, любая позиция $pt_j \in Pt'$ может содержать максимум по одному маркеру каждого из цветов $\overset{5}{\underset{i=1}{c_i}} \in Ct'$. При этом максимальное число маркеров в позиции равно трем (для позиции $pout$), что и служит доказательством ограниченности сети Nt''.

Таким образом, сетью Nt'' порождаются конечные параллельные процессы без альтернатив и конкуренции. Появление в позиции $pout$ одновременно маркеров цветов c_3, c_4 и c_5 (при анализе применимости правила) либо одновременно маркеров цветов c_2 и c_5 (при синтезе дерева по шаблону, задаваемому деревом Tio_k вида (2.19)) соответствует завершению указанных процессов. При этом активизация самого Tio_k как объекта информационного пространства может быть формально определена как достижение тупиковой разметки в сети Nt'' при успешном завершении процесса анализа/синтеза.

Представление анализа входного дерева либо синтеза дерева, получаемого на выходе правила, как процесса, порождаемого сетью Петри, позволяет:

– фиксировать историю процесса анализа применимости правила к дереву расстановкой композиционных меток в узлах для последующего развертывания синтеза дерева, соответствующего выходу правила;

– унифицировать математический аппарат, применяемый для анализа и синтеза дерева в рамках одного и того же сетевого формализма.

Для анализа смысловой взаимной дополняемости глубинных синтаксических структур $Tr_{\chi 1}$ и $Tr_{\chi 2}$ фраз $\chi 1$ и $\chi 2$ в соответствии с *определением 2.5* после анализа применимости правил некоторой заданной Δ-грамматики с построением последовательности преобразований ЛСК требуется сравнить результаты декомпозиции обоих деревьев. Согласно представлению дерева в виде композиции (2.17), здесь выполняется сравнение следующих поддеревьев:

– деревьев, замененных совместной работой лексических правил и обслуживающих их синтаксических замен (обозначим их как $t_1(\chi 1)$ и $t_1(\chi 2)$);

– деревьев верхнего контекста для деревьев $t_1(\chi 1)$ и $t_1(\chi 2)$ (в соответствии с (2.17) это будут $Tr_{01}(\chi 1)$ и $Tr_{01}(\chi 2)$);

– множеств деревьев нижнего контекста для $t_1(\chi 1)$ и $t_1(\chi 2)$ (соответственно, $Tr_{11}(\chi 1), Tr_{21}(\chi 1), \ldots, Tr_{n(\chi 1),1}(\chi 1)$ и $Tr_{11}(\chi 2), Tr_{21}(\chi 2), \ldots, Tr_{n(\chi 2),1}(\chi 2)$).

На основе *определения 2.5* введем понятие функционального соответствия для узлов суммируемых ГСС, представляемых двойками вида (2.15).

Определение 2.8. Будем считать, что узел $w_{\chi 1} \in W_{\chi 1}$ ГСС $Tr_{\chi 1} = \langle W_{\chi 1}, V_{\chi 1} \rangle$ функционально соответствует узлу $w_{\chi 2} \in W_{\chi 2}$ ГСС $Tr_{\chi 2} = \langle W_{\chi 2}, V_{\chi 2} \rangle$, если при описании информационного наполнения этих узлов списками вида (2.16) не будут выполняться следующие условия:

– $(gr_{x1} \neq gr_{x2}) \vee (ar_{x1} \neq ar_{x2}) = true$;

– $(lx_{x1} \neq lx_{x2}) \wedge (lx_{x1} \neq Q) \wedge (lx_{x2} \neq Q) = true$.

Здесь символом Q обозначается нулевая (фиктивная) лексема, см. *определение 2.5*.

Теорема 2.8. Задача установления функционального соответствия деревьев $Tr_{\chi 1}$ и $Tr_{\chi 2}$ принадлежит классу P комбинаторных задач с временной оценкой

n^y, где $n = \max(|W_{\chi 1}|, |W_{\chi 2}|)$, $y = \sum_{i=1}^{|V_R|} \varphi(a_i)$, φ есть матрица вида (1.3), задающая ограничения на характер ветвления в дереве, V_R – словарь пометок на ветвях.

Доказательство теоремы производится через сведение рассматриваемой задачи к известной NP-полной задаче “Изоморфизм подграфу” [19, с. 252].

Заметим, что, как следует из *определения 2.5*, семантическая взаимная дополняемость ЕЯ-фраз на уровне глубинного синтаксиса является относительной. Фактически это означает, что к одной и той же ГСС могут быть применены несколько различных правил преобразования и относительно разных ЛСК. Причем часть из трансформированных и приведенных к виду с единой ЛСК пар глубинных синтаксических структур не подлежит суммированию ввиду функционального несоответствия друг другу согласно *определению 2.8*. Более того, среди ряда допустимых вариантов требуется выбрать пару ГСС, для которой достигается максимум “заполнения мест” в соответствии с *определением 2.6*. Показанная относительность семантической взаимной дополняемости требует рассмотрения функционирования предложенной и исследованной динамической информационной модели системы правил Δ-грамматики в плане:

– активизации взаимно различных информационных элементов применительно к одной и той же ГСС;

– формированием множеств ГСС, ЛФ-синонимичных каждой из суммируемых ГСС при приведении последних к виду с одинаковой ЛСК.

Использование сгенерированных таким образом ЛФ-синони-мических множеств в задаче установления семантической эквивалентности сравниваемых текстов как основной задаче позволяет уйти от неизбежного увеличения затрат памяти ЭВМ и машинного времени для решения основной задачи при использовании предлагаемого метода распознавания семантических повторов. Эти вопросы освещаются в следующем разделе.

2.4. Служебная информация правил и относительность синонимических преобразований деревьев глубинного синтаксиса

Как было показано в разделе 2.1, к одному и тому же дереву глубинного синтаксиса может быть применено несколько правил синонимических замен. В рамках предложенной динамической информационной модели сказанное означает активизацию различных элементов информационного пространства применительно к одной и той же ГСС. Описанная в разделе 2.3 функционально-логическая модель входа/выхода правила Δ-грамматики адекватно отображает различные ситуации его использования как информационного элемента, но не учитывает преобразования, примененные к дереву ранее. В содержательной лингвистической интерпретации это означает невозможность применения правила ко второму и последующему вхождениям заменяемого правилом поддерева в анализируемую ГСС. Сказанное особенно актуально при использовании одних и тех же преобразований как для распознавания сверхфразовых единств в анализируемом тексте, так и при установлении его семантической эквивалентности некоторому другому тексту. В настоящем разделе делается попытка уйти от указанного недостатка предложенной модели путем детализации информации, заносимой при работе правил в анализируемые деревья глубинного синтаксиса.

Действительно, результатом анализа применимости некоторого правила к дереву Tr_{χ} будет заполнение полей $cl_{\chi i}$ в составе четвёрки (2.16) для узлов, выделяемых этим преобразованием. Учитывая возможность применения нескольких правил $\{rule_j,\ldots,rule_k\}\subset\Pi$ синонимических замен к одному и тому же дереву Tr_{χ}, при задании композиционной метки $cl_{\chi i}$ узла следует указывать правило, выделяющее этот узел:

$$cl_{\chi i}=\left(\left(cl_{\chi ij},rule_j\right),\quad\ldots\quad,\left(cl_{\chi ik},rule_k\right)\right), \tag{2.22}$$

а с учетом возможности применения правила к различным частям одного и того же дерева

$$cl_{\chi i} = \left(\left(cl_{\chi ij}, cnt(j), rule_j\right), \quad \ldots \quad , \left(cl_{\chi ik}, cnt(k), rule_k\right)\right), \tag{2.23}$$

где $cnt(j)$ и $cnt(k)$ представляют собой значения счетчика вхождений в дерево Tr_χ поддеревьев, изоморфных тем поддеревьям, которые заменяются правилами $rule_j$ и $rule_k$, соответственно. При этом изоморфизм устанавливается с точностью до функционального соответствия согласно *определению 2.7*.

Аналогично списку (2.22) преобразуется список (2.1):

$$\left\{\left(rule_i, cnt(i), C_0(i, cnt(i))\right) : i = 1, \ldots, |\Pi|\right\}, \tag{2.24}$$

где C_0 есть ключевое слово соответствующей ЛСК.

Список (2.24) формируется в процессе работы сети (2.21) при цветах маркера c_1 и c_3, а элементы списка (2.22) – в ходе следующего прохода той же сети при цвете маркера c_4.

Использование списка (2.23) при анализе применимости правила с расстановкой композиционных меток позволяет избежать зацикливания процесса анализа на одном правиле Δ-грамматики. Действительно, если при цвете маркера c_4 с каждым переходом сети (2.20) связать проверку наличия для узла $w_{\chi i} \in W_\chi$ элементов $\left(cl_{\chi ij}, cnt(j), rule_j\right) \in cl_{\chi i}$, для которых $cl_{\chi ij}$ совпадает с добавляемой композиционной меткой, то повторное выделение в анализируемом дереве Tr_χ одного и того же поддерева, заменяемого одним и тем же правилом, будет невозможно – процесс остановится на вершине заменяемого поддерева.

Формирование списка вида (2.23) для каждого из узлов, выделяемых в дереве Tr_χ, согласуется с формированием списка (2.24) следующим образом.

Элемент списка (2.24), относящийся к некоторому правилу, формируется в случае успешного завершения анализа применимости этого правила и занесения информации в списочной форме (2.23) в поле $cl_{\chi i}$ четвёрко (2.16) для каждого из

выделенных узлов заменяемого правилом поддерева, чему соответствует появление в позиции *pout* сети (2.21) одновременно маркеров цветов c_3, c_4 и c_5.

Если содержать в списке (2.24) информацию только о тех правилах, которые не были применены ранее к дереву, то на случай ложной взаимной дополняемости деревьев $Tr_{\chi 1}$ и $Tr_{\chi 2}$ исключается повторный поиск правил, применимых к указанным деревьям при построении оставшейся части ЛФ-синонимических множеств для $Tr_{\chi 1}$ и $Tr_{\chi 2}$.

Выделяя заменяемые поддеревья по композиционным меткам вида (2.23), можно последовательно относительно разных пар ЛСК определять наличие взаимной дополняемости $Tr_{\chi 1}$ и $Tr_{\chi 2}$ на случай ее отсутствия относительно первой из рассматриваемых пар ЛСК. Тем не менее для корректного взаимодействия процессов увеличения полноты смыслового описания и установления семантической эквивалентности текстов нужно учитывать качественный состав ЛФ-синонимических множеств с точки зрения типов синонимических преобразований, выполняемых при их построении.

Рассмотрим типы преобразований деревьев, допускаемых Δ-грамматикой (1.2) с точки зрения построения целевых выводов, отвечающих требованию обратимости.

Процедура Q_U в составе концептуальной модели (2.3) будет способна строить обратимые выводы, если каждое из используемых ею правил:

– выполняется в обе стороны;

– не ведет к утрате реально выраженных актантов.

Из представленных в [62, с. 152–159] перечня лексических правил первому требованию не отвечают смысловые импликации (правила № 49–56). Лексические правила № 7, 8 и 9 выполняются в обе стороны, однако их применение процедурой Q_U исключено ввиду того, что описываемые ими конверсивные замены ведут к утрате места (валентности) в перерабатываемой ГСС. Кор-

ректное применение указанных правил возможно лишь тогда, когда отпадающая валентность в перерабатываемой ГСС не была заполнена.

Пусть $\Pi_{LSC} \subset \Pi$ есть множество правил Δ-грамматики (1.2), удовлетворяющих вышеуказанным требованиям.

Теорема 2.9. Построение обратимых выводов процедурой Q_U возможно только с применением правил из множества Π_{LSC}.

Доказательство теоремы естественным образом вытекает из рассмотренных в разделе 2.2 свойств языка сети, моделирующей систему правил Δ-грамматики. При ограничении Δ-грамматикой (1.2) рассмотрением правил множества Π_{LSC} любое слово в языке указанной сети будет обратимым.

Таким образом, при выделении сверхфразовых единств на множестве деревьев глубинного синтаксиса в соответствии с *определением 2.6* следует использовать правила множества Π_{LSC}.

Обозначим множества, порождаемые Δ-грамматикой (1.2) для деревьев $Tr_{\chi 1}$ и $Tr_{\chi 2}$ применением правил из Π_{LSC} относительно некоторого фиксированного ключевого слова, как $TLSC_{\chi 1}$ и $TLSC_{\chi 2}$, соответственно. Тогда в случае отсутствия пары деревьев $Tr_{\chi 1 i} \in TLSC_{\chi 1}$ и $Tr_{\chi 2 j} \in TLSC_{\chi 2}$, для которых возможно построение формального образа сверхфразового единства в соответствии с *определением 2.6*, впоследствии, уже в процессе установления эквивалентности каждой из фраз $\chi 1$ и $\chi 2$ заданному эталону, в множества $TLSC_{\chi 1}$ и $TLSC_{\chi 2}$ будут заноситься деревья, получаемые из $Tr_{\chi 1}$ и $Tr_{\chi 2}$ применением правил $rule \in \Pi_{LSC}$, упоминаемых в списках вида (2.24) для $Tr_{\chi 1}$ и $Tr_{\chi 2}$, соответственно, и не использованных при приведении этих деревьев к виду с одинаковой ЛСК. А поскольку перестройке подлежит только заменяемое правилом поддерево, то композиционные метки, расставляемые в дереве другими правилами множества Π_{LSC}, будут сохранены. Без изменения также остаются соответствующие

элементы списков (2.24) для деревьев из множеств $TLSC_{\chi 1}$ и $TLSC_{\chi 2}$. Применение списков (2.24) и композиционных меток (2.23) таким образом позволяет избежать полного просмотра ЛФ-синонимических множеств при определении возможности построения рассматриваемой Δ-грамматикой очередного дерева.

2.5. Пример построения образа сверхфразового единства для четырех простых распространенных предложений русского языка

Рассмотрим работу предложенного механизма распознавания сверхфразовых единств на примере высказывания из четырех простых распространенных предложений русского языка:

1) “*Лаборатория провела эксперименты по изучению условных рефлексов*”;

2) “*Подопытными животными были собаки*”;

3) “*Результаты экспериментов рассматривались в докладе на конференции*”;

4) “*Ученый детально анализировал результаты проведенных опытов*”.

С целью более наглядной демонстрации применения основных идей настоящей главы исходные предложения построены на основе лексики, описанной в Толково-комбинаторном словаре современного русского языка [162].

Скобочное описание дерева глубинного синтаксиса первого предложения с использованием списков вида (2.16), представленное на рис. 2.3, *б*, уже было затронуто нами в разделе 2.3. Аналогичные описания глубинных синтаксических структур для второго, третьего и четвертого предложений представлены на рис. 2.6.

```
(
 (( Экспериментировать  A2  Oper1 )
    (V (несов_вид прош_вр изъявит_накл )) 0 nil
 )
   ((( Собака nil )( S (мн_ч)) 1 nil ))
   ((( Экспериментировать  //A2 )( S (мн_ч)) 2 nil )
    ((( Q nil ) nil 1 nil ))
    ((( Q nil ) nil 2 nil ))
   )
)
```

а

```
  (
   (( Рассматривать nil )
      (V (несов_вид прош_вр изъявит_накл )) 0 nil
   )
     ((( Q nil )( S nil ) 1 nil ))
     ((( Экспериментировать S0 Sres )( S (мн_ч)) 2 nil )
      ((( Экспериментировать S0 )( S (мн_ч)) 2 nil )
        ((( Q nil ) ( S nil ) 1 nil ))
        ((( Q nil ) ( S ( на nil )) 2 nil ))
        ((( Q nil ) ( S ( по nil )) 3 nil ))
      )
     )
     ((( Доклад nil )( S (в ед_ч)) 3 nil )
      ((( Конференция nil )( S ( на nil )) 2 nil ))
     )
  )
```

б

```
(
 (( Рассматривать Syn )
    (V (несов_вид прош_вр изъявит_накл )) 0 nil
 )
   ((( Ученый nil )( S (ед_ч)) 1 nil ))
   ((( Экспериментировать S0 Sres )( S (мн_ч)) 2 nil )
    ((( Экспериментировать S0 Syn )( S (мн_ч)) 2 nil )
      ((( Q nil )( S nil ) 1 nil ))
      ((( Q nil )( S (на nil)) 2 nil ))
      ((( Q nil )( S (по nil)) 3 nil ))
      ((( Проведенный nil )( P (мн_ч)) 5 nil ))
    )
   )
   ((( Q nil )( S (в nil)) 3 nil ))
   ((( Рассматривать Magn[‘аспекты’] )( Adv nil ) 5 nil ))
)
```

в

Рис. 2.6. Анализируемые деревья глубинного синтаксиса:
а – второго предложения; *б* – третьего предложения; *в* – четвертого предложения

Определяя применимость лексических синонимических преобразований, описанных в [62, с. 152–159] и отвечающих *теореме 2.9*, для ГСС исходных предложений формируем списки (2.24), представленные в табл. 2.3.

Таблица 2.3

Применимость лексических синонимических преобразований для исходных предложений

№ предложения	Результат анализа применимости
1	((17 1 Экспериментировать))
2	((16 1 Экспериментировать))
3	((1 1 Рассматривать)(1 2 Экспериментировать))
4	((1 1 Рассматривать)(1 2 Экспериментировать))

К первому предложению применимо лексическое правило № 17 с обслуживающим его синтаксическим правилом № 6, [62, с. 154]. Заметим, что условие применимости данного правила, касающееся грамматических характеристик ключевого слова (C_0 – глагол), уже заложено в соответствующий компонент списочного описания (2.16) вершины выходного дерева и представлено символьным обозначением глагола из табл. 2.1. Соответствующий переход в сети Петри, моделирующей рассматриваемую систему правил, является безусловным.

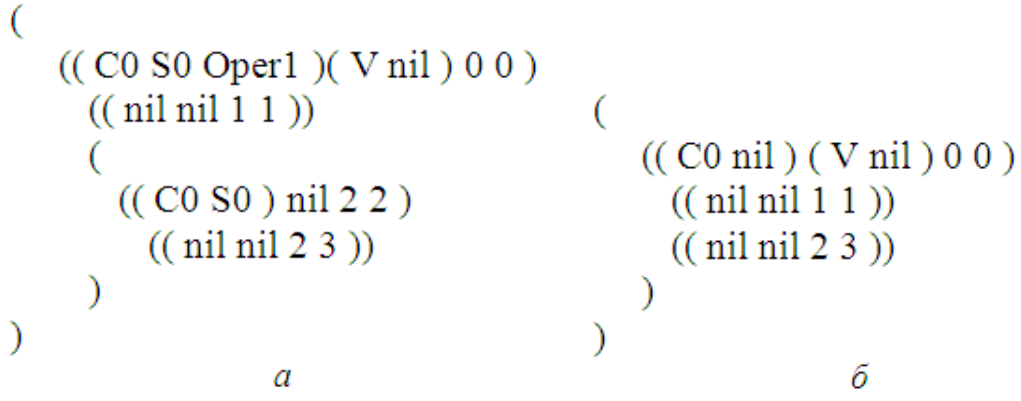


Рис. 2.7. Лексическое правило № 17 из представленных в [62, с. 152–159]:
а – списочное описание входного; *б* – выходного дерева

Ко второму предложению применимо лексическое правило № 16 с обслуживающим его синтаксическим правилом № 8, [62, с. 153]. Как и в предыдущем случае, условие применимости в виде логической формулы (2.6) отдельно не выносится и заложено в описании выходного дерева правила (рис. 2.8, *б*). Для обоих предложений лексико-синтаксические замены рассматриваются относительно ключевого слова C_0 = *"Экспериментировать"*.

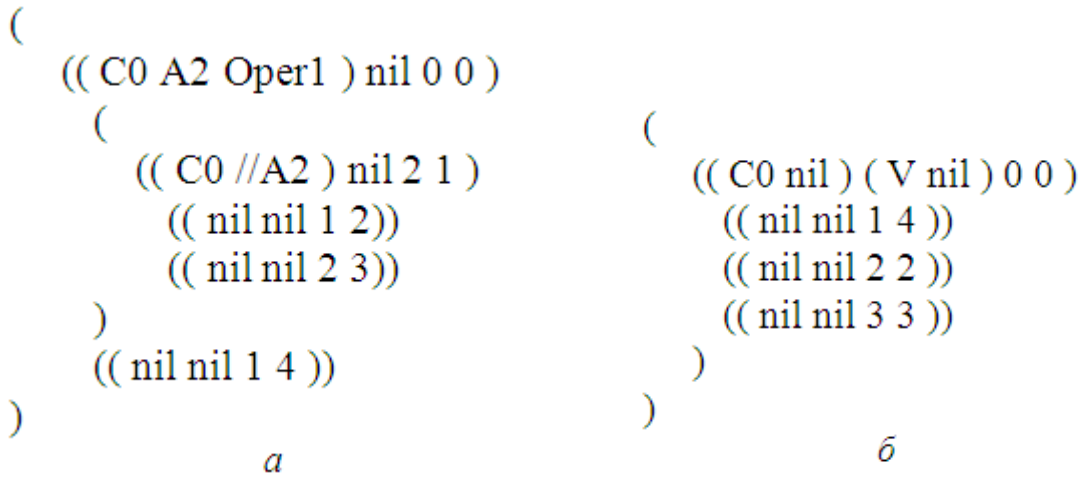


Рис. 2.8. Лексическое правило № 16 из представленных в [62, с. 152–159]:
а – списочное описание входного; *б* – выходного дерева

Посредством применения указанных правил оба предложения приводятся к виду с одинаковой ЛСК относительно ключевого слова C_0 = *"Экспериментировать"*.

1) “*Лаборатория экспериментировала на* S_{nn} *с целью изучения условных рефлексов*”;

2) “*Экспериментировал (а,о,и) на собаках*”.

Преобразованные деревья глубинного синтаксиса первого и второго предложения в скобочной нотации представлены на рис. 2.9. Заполняя незаполненные места глубинно-синтаксических актантов в соответствии с *определением 2.6*, получаем формальный образ сверхфразового единства для первого и второго предложений в виде ГСС на рис. 2.10.

```
(
 (( Экспериментировать  nil )
    (V (несов_вид прош_вр изъявит_накл )) 0 nil
 )
   ((( Лаборатория nil )( S (ед_ч)) 1 nil ))
   ((( Q nil )( S (на мн_ч)) 2 nil ))
   ((( Изучение nil )( S (по ед_ч)) 3 nil )
    ((( Рефлекс nil )( S (мн_ч)) 2 nil )
      ((( Условный nil )( A (мн_ч)) 5 nil )
    )
   )
)
```

а

```
(
 (( Экспериментировать  nil )
    (V (несов_вид прош_вр изъявит_накл )) 0 nil
 )
   ((( Q nil )( S (ед_ч)) 1 nil ))
   ((( Собака nil )( S (на мн_ч)) 2 nil ))
   ((( Q nil )( S (по ед_ч)) 3 nil ))
)
```

б

Рис. 2.9. Преобразованные деревья относительно C_0 = *"Экспериментировать"*: *а* – первого предложения; *б* – второго предложения

```
(
 (( Экспериментировать  nil )
    (V (несов_вид прош_вр изъявит_накл )) 0 nil
 )
   ((( Лаборатория nil )( S (ед_ч)) 1 nil ))
   ((( Собака nil )( S (на мн_ч)) 2 nil ))
   ((( Изучение nil )( S (по ед_ч)) 3 nil )
    ((( Рефлекс nil )( S (мн_ч)) 2 nil )
      ((( Условный nil )( A (мн_ч)) 5 nil )
    )
   )
)
```

Рис. 2.10. Суммарная ГСС для первого и второго предложения

К дереву глубинного синтаксиса третьего предложения дважды применимо лексическое правило № 1, [62, с. 152], вход и выход которого в принятой нами скобочной нотации описывается как

(((C0 nil) nil 0 0)) и (((C0 Syn) nil 0 0)),

соответственно. Это же правило, но в обратном направлении, применимы к ГСС четвертого предложения. Посредством применения первого вхождения указанного правила относительно ключевого слова C_0="*Рассматривать*" приводим ГСС обоих предложений к виду с одинаковой ЛСК. При этом дерево ГСС третьего предложения остается без изменений, а ГСС четвертого предложения приводится к виду, представленному на рис. 2.11.

```
(
 (( Рассматривать  nil )
    (V (несов_вид прош_вр изъявит_накл )) 0 nil
 )
   ((( Ученый nil )( S (ед_ч)) 1 nil ))
   ((( Экспериментировать  S0 Sres )( S (мн_ч)) 2 nil )
      ((( Экспериментировать  S0 Syn )( S (мн_ч)) 2 nil )
         ((( Q nil )( S nil ) 1 nil ))
         ((( Q nil )( S (на nil)) 2 nil ))
         ((( Q nil )( S (по nil)) 3 nil ))
         ((( Проведенный nil )( P (мн_ч)) 5 nil ))
      )
   )
   ((( Q nil )( S (в nil)) 3 nil ))
   ((( Рассматривать  Magn[‘аспекты’] )( Adv nil) 5 nil ))
)
```

Рис. 2.11. Преобразованное дерево четвертого предложения относительно C_0="*Рассматривать*"

Тем не менее, дерево глубинного синтаксиса третьего предложения (рис. 2.6, *б*) и преобразованная глубинная синтаксическая структура четвертого предложения (рис. 2.11) не могут функционально соответствовать друг другу по *определению 2.8* в силу наличия синонима для C_0="*Экспериментировать*".

```
(
 (( Рассматривать  nil )
    (V (несов_вид прош_вр изъявит_накл )) 0 nil
 )
   ((( Ученый nil )( S (ед_ч)) 1 nil ))
   ((( Экспериментировать  S0 Sres )( S (мн_ч)) 2 nil )
      ((( Экспериментировать  S0 )( S (мн_ч)) 2 nil )
         ((( Q nil )( S nil ) 1 nil ))
         ((( Q nil )( S (на nil)) 2 nil ))
         ((( Q nil )( S (по nil)) 3 nil ))
         ((( Проведенный nil )( P (мн_ч)) 5 nil ))
      )
   )
   ((( Q nil )( S (в nil)) 3 nil ))
   ((( Рассматривать  Magn[‘аспекты’] )( Adv nil) 5 nil ))
)
```

Рис. 2.12. Окончательный вариант дерева четвертого предложения после замены синонима для C_0="*Экспериментировать*"

```
(
 (( Рассматривать  nil )
    (V (несов_вид прош_вр изъявит_накл )) 0 nil
 )
   ((( Ученый nil )( S (ед_ч)) 1 nil ))
   ((( Экспериментировать  S0 Sres )( S (мн_ч)) 2 nil )
      ((( Экспериментировать  S0 )( S (мн_ч)) 2 nil )
         ((( Q nil )( S nil ) 1 nil ))
         ((( Q nil )( S (на nil)) 2 nil ))
         ((( Q nil )( S (по nil)) 3 nil ))
         ((( Проведенный nil )( P (мн_ч)) 5 nil ))
      )
   )
   ((( Доклад nil )( S (в ед_ч)) 3 nil ))
   ((( Рассматривать  Magn[‘аспекты’] )( Adv nil) 5 nil ))
)
```

Рис. 2.13. Суммарная ГСС для третьего и четвертого предложения

Указанное несоответствие устраняется применением второго правила из списка (2.24), представленного в табл. 2.3 для четвертого предложения. При этом дерево глубинного синтаксиса четвертого предложения преобразуется к виду на рис. 2.12. Формальный образ сверхфразового единства для третьего и четвертого предложения представлен на рис. 2.13.

Далее рассматриваем возможность суммирования деревьев глубинного синтаксиса, представленных на рис. 2.10 и 2.13. Аналогично деревьям дискретных предложений определяем применимость лексических синонимических преобразований, описанных в [62, с. 152–159] и отвечающих *теореме 2.9*, для указанных глубинных синтаксических структур с формированием списков вида (2.24). Результаты представлены в табл. 2.4.

Таблица 2.4

Применимость лексических синонимических преобразований для суммарных ГСС

№ предложений	Результат анализа применимости
1 и 2	((17 1 Экспериментировать))
3 и 4	((1 1 Рассматривать)(17 1 Рассматривать) (1 2 Экспериментировать))

Как видно из табл. 2.4, единственным ключевым словом, относительно которого возможно приведение суммарных ГСС к виду с одинаковой ЛСК, является C_0=*"Экспериментировать"*. Однако на основе начального сценария, соответствующего активизации входов/выходов лексических правил № 1 и № 17, требуемую последовательность преобразований в рассматриваемой системе правил построить нельзя. Поэтому предложенный механизм распознавания сверхфразовых единств для рассмотренного примера завершает свою работу, выдав в качестве окончательного результата деревья, представленные на рис. 2.10 и 2.13.

Выводы

Таким образом, во второй главе предложены теоретические основы сжатия информации для прецедентов классов СЭ уровня абстрактной лексики.

Предложенный в главе подход к построению совокупности целевых выводов в Δ-грамматике позволяет теоретически обосновать принципиальную возможность существования алгоритмического решения для задач сравнения помеченных деревьев, требующих качественного анализа представленной в деревьях информации. Применение данного подхода в задаче сжатия смысловой информации на уровне глубинного синтаксиса позволяет выделять семантические повторы в анализируемом тексте без существенного ограничения его жанра, в то время как большинство из известных алгоритмически разрешимых методов распознавания сверхфразовых единств ориентированы на тексты определенного жанра. При этом динамическая информационная модель системы правил Δ-грамматики сводит поиск последовательности преобразований с заданными свойствами к классическим задачам теории сетей Петри.

Тем не менее, при практической реализации предложенного подхода актуальна проблема автоматизации накопления знаний об описываемых логическими формулами (2.6) условиях применимости правил синонимических преобразований помеченных деревьев. В частности, требуется рассмотреть вопросы формализации толкования лексического значения слова, представляемого на естественном языке в специализированном толковом словаре, с целью автоматизированного получения и систематизации указанных знаний.

Решению данной задачи на основе идей и методов АФП посвящается третья глава диссертационной работы.

Глава 3
ФОРМИРОВАНИЕ И ЭКСПЕРИМЕНТАЛЬНАЯ ОЦЕНКА ЗНАНИЙ НА ОСНОВЕ СИТУАЦИЙ СМЫСЛОВОЙ ЭКВИВАЛЕНТНОСТИ

Настоящая глава посвящена использованию СЭ-текстов в качестве исходных данных формирования и классификации семантических отношений как основы знаний о синонимии. Введено понятие прецедента ЛФ-синонимии, основанное на смысловых отношениях в рамках стандартных ЛФ. Решена задача автоматизации накопления и экспериментальной оценки знаний об условиях применимости синонимических преобразований деревьев глубинного синтаксиса. Предложено формализованное средствами логики предикатов первого порядка описание толкования лексического значения слова. Исследованы принципы обобщения независимых вариантов толкований слова относительно предметно-ограниченного подмножества ЕЯ. Сформулирован и теоретически обоснован принцип формирования и кластеризации семантических отношений выделением синтагматических зависимостей на множествах СЭ-фраз. Основные результаты главы представлены в [32, 73–75, 80, 83, 87–89, 113, 143, 149, 150, 164–166, 177, 178].

3.1. Лексическое значение слова и его формализация на языке логики предикатов первого порядка

В рамках рассмотренного нами подхода “Смысл⇔Текст” большинство словарных единиц языка возникает при переходе от семантического представления к глубинным синтаксическим структурам. Фрагмент семантического представления, который соответствует отдельному ЕЯ-слову, представляет собой толкование лексического значения (ЛЗ) этого слова. В работе [3] Ю.Д. Апресян исследует связь между толкованием слова и его МУ для решения задачи построения глубинной синтаксической структуры по фрагменту семантического представления. Цель настоящего раздела состоит в том, чтобы показать связь

между толкованием ЛЗ слова и его смыслом, актуальную для формирования прецедента класса СЭ.

Как уже было показано в разделе 1.3, прецеденту класса СЭ на верхнем уровне иерархии знаний о синонимии соответствует условие применимости некоторого правила синонимического преобразования глубинных синтаксических структур. Данное условие выполняет функцию фильтра, который запрещает синтез ЕЯ-фразы из множества семантически эквивалентных, если конечный продукт синтеза дает нарушение лексического значения, сочетаемости или стилистических норм. Многие фильтры были описаны в работах И.А. Мельчука и А.К. Жолковского, упомянутых в [62]. Однако, как отметил академик Ю.Д. Апресян [3, с. 336], проблема нуждается в дальнейшей разработке. Тем более что, по оценке И.А. Мельчука [62, с. 159–160], специальных исследований по данному вопросу не проводилось, а сами правила описаны в первом приближении.

Следует отметить, что метод фильтров является традиционным методом построения синтаксической структуры фразы русского языка. Как показано в [106], его применение предполагает установление для большинства слов нескольких потенциально возможных связей с различными управляющими словами. Роль фильтров при этом состоит в выборе правильных вариантов анализа. Одним из подходов к решению задачи выбора корректного варианта здесь является привлечение семантической информации из словаря. Важнейшую роль при этом играет информация о семантической интерпретации глубинных синтаксических актантов предикатного слова, описываемая его моделью управления. Тем не менее при наличии у слова более одного ЛЗ становятся возможными альтернативные варианты разбиения анализируемой ЕЯ-фразы на словосочетания (именные группы (ИГ)), каждый из которых удовлетворяет требованию фильтров. В частности, для предикатных слов с каждым ЛЗ связывается альтернативный вариант МУ и соответствующий синоним с более широким, чем у самого слова, значением. При синонимическом преобразовании исходной ЕЯ-фразы на уровне глубинного синтаксиса названный фактор может привести к построению неадекватных перифраз.

Наиболее естественный путь решения показанной проблемы заключается в привлечении информации словарных определений (дефиниций) [3, 139] для тех понятий, которые обозначаются актантами предикатного слова. При этом введение в рассмотрение аналогичных определений для семантики произвольных отношений, отличных от связей предиката с актантами по МУ и задаваемых входящими в анализируемое предложение именными группами, позволяет более точно устанавливать соответствия требованиям семантической интерпретации глубинных синтаксических актантов предикатного слова при построении дерева ГСС. Данная точка зрения естественным образом согласуется со сформулированным в разделе 1.4 определением прецедента класса СЭ. При этом исходными данными формирования условия применимости правила будут признаки слов в парах ЕЯ-высказываний, сравниваемых по смыслу. Далее в настоящей главе мы рассмотрим, каким образом данная информация может быть выявлена на основе лексикографического толкования слова.

В работе [139] на примере генитивной конструкции русского языка исследуется взаимодействие формальной и лексической семантики в задаче построения формализованного описания значения слова. Представляемая Б.Х. Парти и В.Б. Борщевым идея состоит в выделении совокупности свойств обозначаемого словом объекта реального мира и последующем описании ЛЗ слова посредством теории – совокупности аксиом (meaning postulates), каждая из которых описывает отдельное свойство этого объекта. Само задаваемое посредством набора аксиом описание ЛЗ слова здесь соответствует теории сорта обозначаемой словом реалии. При этом понятие сорта как элемента “наивной картины мира” и класса, к которому язык относит конкретную реалию, фактически соответствует тому, что в публикациях Московской лингвистической школы, в частности в монографиях [3] и [62], понимается под семантическим классом (СК) обозначающего эту реалию слова. Такое же понимание СК использовалось и в [80] относительно описания семантической интерпретации глубинного синтаксического актанта предикатного слова. Для описания самой теории сорта в [139] используется принятое в

формальной семантике λ-выражение (выражение с оператором абстракции лямбда [13]), которое возвращает в качестве значения множество всех объектов, принадлежащих заданному сорту.

Рассмотрим вначале ряд свойств формализованного описания лексического значения слова в виде теории, которые необходимо принять во внимание при программной реализации соответствующего компонента словарной базы знаний.

Во-первых, указанное представление, используемое в [139] для определения сортов опорных существительных именных групп, есть описание свойств объектов, принадлежащих некоторому сорту. Фактически это означает, что из всех возможных отношений, задаваемых именными группами и связываемых с лексическими значениями их опорных слов, первоочередную значимость для нас имеют лексические отношения – те отношения, которые задаются самими опорными словами.

Во-вторых, вводится оператор типового сдвига для преобразования унарных отношений типа $\langle e,t \rangle$ [3], которые исходно сопоставляются словарным значениям опорных слов именных групп, в задаваемые этими ИГ бинарные отношения (пример – метонимический сдвиг слова с ЛЗ *“контейнер”* в сорт *“квант”*, описанный в [139]). Введение такого оператора требует формального описания уже не теории сорта, а задаваемого этим сортом отношения. При этом и имя отношения (как имя сорта), и его аргументы представляются аргументами функции – λ-выражения, сопоставляемого именной группе. Здесь следует отметить, что имя отношения, определяемого сортом опорного слова ИГ, как и сам этот сорт, в терминологии Московской лингвистической школы следует отождествлять с семантическим классом, но не отдельного слова, а всего словосочетания именной группы. Так, для глагола “*сжечь*” в значении “*израсходовать*” СК актанта количественной ролевой ориентации (*“Quant”*) соответствует именно количественно-

[3] Здесь имеется в виду используемое в формальной семантике понятие “тип”, *e* и *t* соответствуют элементарным типам – сущностям и формулам.

му отношению (*“Quant”, “квант”*), которое задается рассмотренной в [139] генитивной конструкцией меры (пример – “*сжечь машину дров*”).

В-третьих, в концептуальном плане теория лексического значения слова представляется набором утверждений, связывающих его с другими словами (в первую очередь здесь рассматривается связь между обозначаемыми словами понятиями). Отдельное утверждение теории описывает бинарное отношение между некоторыми известными понятиями. Каждое из понятий, выступающих в роли аргументов отношения, по сути, соответствует одному из известных СК. Имя самого отношения задается ЕЯ-словом, для которого явным образом в словарной базе знаний указан семантический класс обозначаемой этим словом сущности.

В работе [139] в качестве аргументов функции, описывающей задаваемое генитивной конструкцией отношение, выступают элементы конструкции – опорное слово и генитивная группа (зависимое слово). Но, рассуждая о приемлемости той или иной генитивной конструкции, принято говорить не о входящих в нее словах, а о сортах обозначаемых этими словами реалий. Исходя из этого соображения, в настоящей работе теорию отношения, определяемого ИГ, мы будем рассматривать не относительно самих слов-элементов именной группы, а относительно соответствующих им семантических классов.

На основе вышесказанного, а также в соответствии со сформулированной нами *задачей 1.2*, представим описание теории ЛЗ слова w_i, заменяемого посредством некоторого правила $rule_j \in \Pi$, в виде двойки:

$$Lm(w_i) = (w_i, LM), \tag{3.1}$$

где элементы списка LM задают отношения между словами и понятиями. Отдельный элемент списка LM может представлять как бинарное отношение между парой понятий $\{o_1, o_2\} \subset O$:

$$Mp = (r_2, o_1, o_2), \tag{3.2}$$

так и рекурсивно определяемое отношение произвольной арности вида

$$Mp = (r_n, o, LM_r), \tag{3.3}$$

либо

$$Mp = (r_c, LM_r), \tag{3.4}$$

где $r_c \in \{\vee, \&, \neg\}$; r_2 и r_n – символы (либо символьные цепочки), обозначающие соответствующие отношения; список LM_r определяется по аналогии с LM. Посредством LM_r в формуле (3.3) задается связь понятия o с другими словами и понятиями. На месте обозначений понятий и отношений в утверждениях (3.2)–(3.4) из состава описания теории ЛЗ могут быть переменные.

Сама теория ЛЗ слова, задаваемая посредством двойки вида (3.1), может быть представлена составным объектом языка Пролог, в свою очередь легко преобразуемым в структуры специализированного домена *tree* для работы с деревьями в Visual Prolog'e.

На рис. 3.1 приведены древовидные описания теорий для ЛЗ слов "*эксперимент*" и "*экспериментировать*", упоминавшихся в примере из раздела 2.5. Исходные варианты толкований взяты из Толково-комбинаторного словаря современного русского языка И.А. Мельчука и А.К. Жолковского [162].

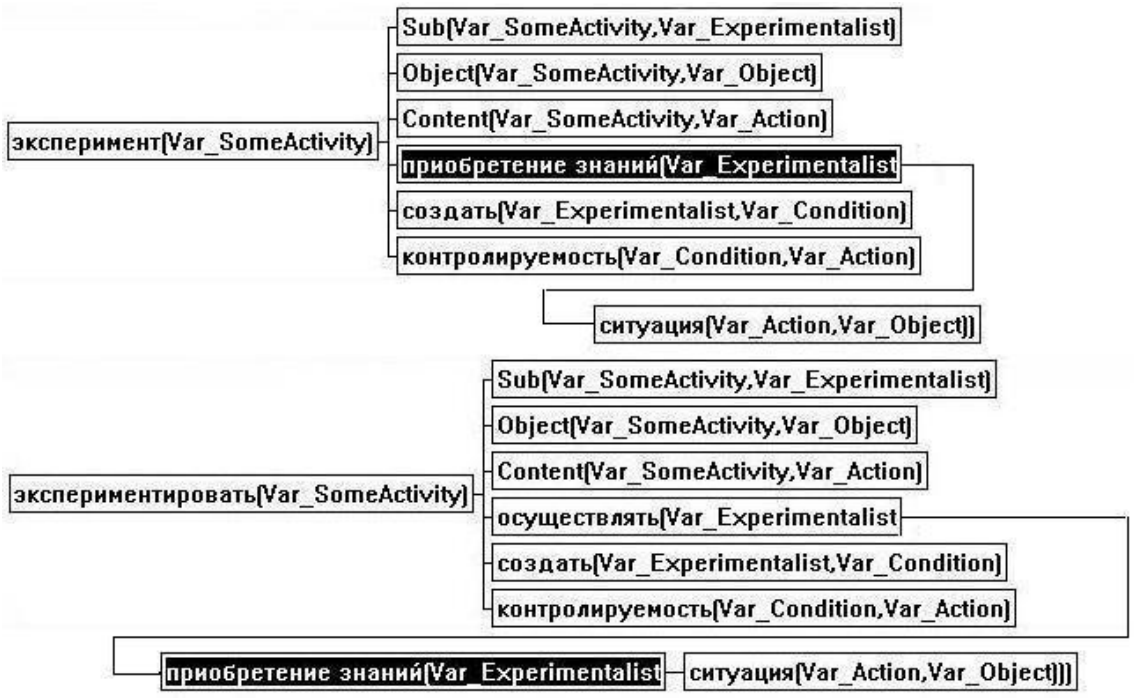


Рис. 3.1. Теории ЛЗ "эксперимент" и "экспериментировать"

Утверждение 3.1. Если имеется формализованное описание теории $Lm(w_i) = (w_i, LM)$ ЛЗ слова w_i, задаваемое формулой вида (3.1), то смысл этого слова определяется набором характеристических функций (ХФ) ChF_{hi} таких, что выполняются следующие условия:

1. В списке LM содержится тройка $Mp = (r_2, o_1, o_2)$ вида (3.2) (обозначим ее как ChF_{Val}), при этом $ChF_{hi}(w_i) = o_2$, где o_2 – обозначение известного системе понятия (семантического класса). При этом в роли списка LM может выступать список LM_r в составе некоторой тройки вида (3.3).

2. Существует тройка (далее обозначаемая как ChF_{Name}) либо вида (3.2), либо вида (3.3), но в обоих случаях ChF_{hi} является первым её элементом и представляет имя известного смыслового (семантического) отношения.

3. Если ChF_{Name} есть первая тройка, удовлетворяющая условию (2) при обратном просмотре списка LM от ChF_{Val}, и существует подсписок LM' списка LM, при этом, соответственно, либо $LM' = \{(ChF_{hi}, o_{11}, o_{12}), \ldots, ChF_{Val}\}$, либо $LM' = \{(ChF_{hi}, o, LM_r) : ChF_{Val} \in LM_r\}$, а каждое последующее утверждение в LM' должно иметь как минимум один общий аргумент, являющийся обозначением некоторой переменной, с предыдущим утверждением.

В качестве примера на рис. 3.2 представлен вариант теории для ЛЗ слова "*агрессор*", а на рис. 3.3 – соответствующий ему набор характеристических функций. Как и на рис. 3.1, исходный вариант толкования взят в [162].

Рис. 3.2. Анализируемый вариант теории ЛЗ

Рис. 3.3. Характеристические функции и формальные признаки их значений

В приведённых примерах *Var_SomeBody* обозначает переменную для слова, интерпретируемого посредством формализованного описания (3.1) теории ЛЗ. Согласно *условию 2 утверждения 3.1*, эта же переменная будет вторым элементом в тройке ChF_{Name}.

Фактически посредством *утверждения 3.1* мы сформулировали точное определение смысла слова на основе *определения 1.6*, более близкого пониманию смысла в философской логике. Опираясь на понятия экстенсионала и интенсионала, рассмотрим решение задачи обобщения знаний, представляемых формулами вида (3.1), на основе математических методов АФП. Данная задача актуальна при независимом построении теории слова разными исследователями, в частности при построении теорий на основе ЕЯ-толкований с применением стандартных концептуальных языков [33, 124, 125, 152].

Представим систему элементов толкования заданного слова для независимых вариантов теории лексического значения посредством многозначного формального контекста следующего вида[4]:

$$Klm = (Glm, Mlm, Vlm, Ilm), \tag{3.5}$$

где $\forall\, glm \in Glm$ есть некоторый вариант толкования ЛЗ слова w_i в форме (3.1). Множество признаков $Mlm = Mlm_1 \cup Mlm_2$, при этом если $mlm \in Mlm_1$, то $mlm = ChF_{hi}(w_i)$, а если $mlm \in Mlm_2$, то mlm – это имя некоторого известного СК или отношения, выступающего к качестве первого элемента тройки вида (3.2) в составе списка LM', формируемого согласно условию (3) *утверждения 3.1*.

[4] В обозначениях из формалы (3.5) "*lm*" есть сокр. от англ. lexical meaning – лексическое значение.

Множество признаковых значений $Vlm = Vlm_1 \cup Vlm_2$, при этом если $vlm \in Vlm_1$, то vlm есть имя ХФ ChF_{hi}, для которой задано $ChF_{hi}(w_i)$. Если же $vlm \in Vlm_2$, то vlm есть значение ХФ для некоторого $w_{1i} \neq w_i$: $Lm(w_{1i}) = (w_{1i}, LM')$, а сам LM': $LM' \subset LM$ формируется согласно условию (3) *утверждения 3.1*. Тернарное отношение Ilm задает частичное отображение Glm на Vlm: $mlm(glm) = vlm$, содержательно – ставит в соответствие каждой ХФ её значение для заданного w_i.

Лексическое значение слова, описываемое посредством формализованной теории (3.1), есть денотация. В логике ей ставится в соответствие экстенсионал как класс сущностей, которые определяются посредством теории. При этом внешне различные описания теорий одного и того же ЛЗ определяют единое множество характеристических функций, задаваемых в соответствии с *утверждением 3.1*. Характеристические функции (в том числе определяемые рекурсивно для списков в составе троек вида (3.3) и двоек вида (3.4)) задают набор формальных признаков для элементов толкования лексического значения. В конечном итоге они определяют интенсионал обобщенной теории заданного лексического значения.

Таким образом, исходя из определения интенсионала как функции от возможных миров к экстенсионалам, а также рекурсивной природы постулатов значения, имеем *задачу* построения обобщенной теории лексического значения как восстановление синтаксического представления экстенсионала на основе известного синтаксиса λ-выражений для ХФ, составляющих интенсионал.

Утверждение 3.2. Утверждения (r_n, o, LM_1) и (r_n, o, LM_2) вида (3.3) могут быть представлены одним “ИЛИ”-утверждением $(r_n, o, \{("or", LM_3)\})$, если наборы ФП, полученные на основе LM_1, LM_2 и LM_3, в решётке ФП для формального контекста вида (3.5) образуют области $\Re lm(Glm_1, Mlm_1, Vlm_1, Ilm)$, $\Re lm(Glm_2, Mlm_2, Vlm_1, Ilm)$ и, соответственно, $\Re lm(Glm_3, Mlm_3, Vlm_1, Ilm)$ с НОСП, которое имеет r_n в качестве значения признака. При этом:

$$Glm_1 = \{(w_{1i}, LM_1)\},\ Glm_2 = \{(w_{2i}, LM_2)\},\ Mlm_1 \neq Mlm_2,\ Mlm_3 = Mlm_1 \cup Mlm_2,$$

$$\Re lm(Glm_3, Mlm_3, Vlm_1, Ilm) = \Re lm(Glm_1, Mlm_1, Vlm_1, Ilm) \cup \Re lm(Glm_2, Mlm_2, Vlm_1, Ilm).$$

Компоненты w_{1i} и w_{2i} в вышеуказанном соотношении есть некоторые символьные цепочки, не обязательно являющиеся ЕЯ-словами. Их примером могут послужить "*Толкование2_агрессор*" и "*Толкование3_агрессор*" на рис. 3.4.

Утверждение 3.3. Утверждения (r_n, o, LM_1) и (r_n, o, LM_2) вида (3.3) могут быть представлены одним "И"-утверждением $(r_n, o, \{("and", LM_3)\})$, если на основе LM_1, LM_2 и LM_3 определяются формальные понятия (X, Y_1), (X, Y_2) и (X, Y_3) в решётке для формального контекста (3.5), при этом $Y_3 = Y_1 \cup Y_2$.

Замечание. Согласно *утверждению 3.1* внешне различные описания теорий вида (3.1) для одного и того же ЛЗ задают единое множество характеристических функций. Следовательно, мощность указанного множества не зависит от количества обобщаемых теорий. Временная оценка процесса обобщения теорий для заданного ЛЗ составляет $\left(\frac{n}{k}\right)^k$, где n – мощность множества ХФ, k – количество обобщаемых теорий. Поскольку $k \in [1, \ldots, n]$, то $\left(\frac{n}{k}\right)^k = n$ при $k = 1$ и $\left(\frac{n}{k}\right)^k = 1$ при $k = n$. В худшем случае n равно числу утверждений вида (3.2) и (3.3) на всех уровнях формализованного описания (3.1).

В качестве примера на рис. 3.4 представлена решетка ФП для трех вариантов толкования ЛЗ "*агрессор*", а на рис. 3.5 – результат их обобщения.

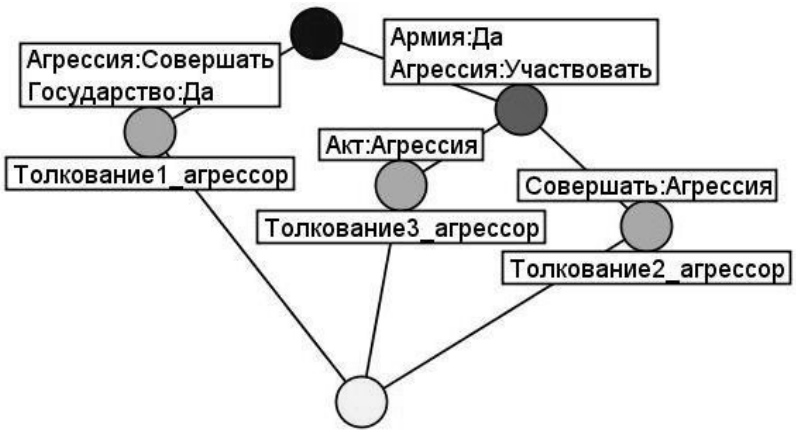


Рис. 3.4. Формализованные толкования для ЛЗ *"агрессор"*

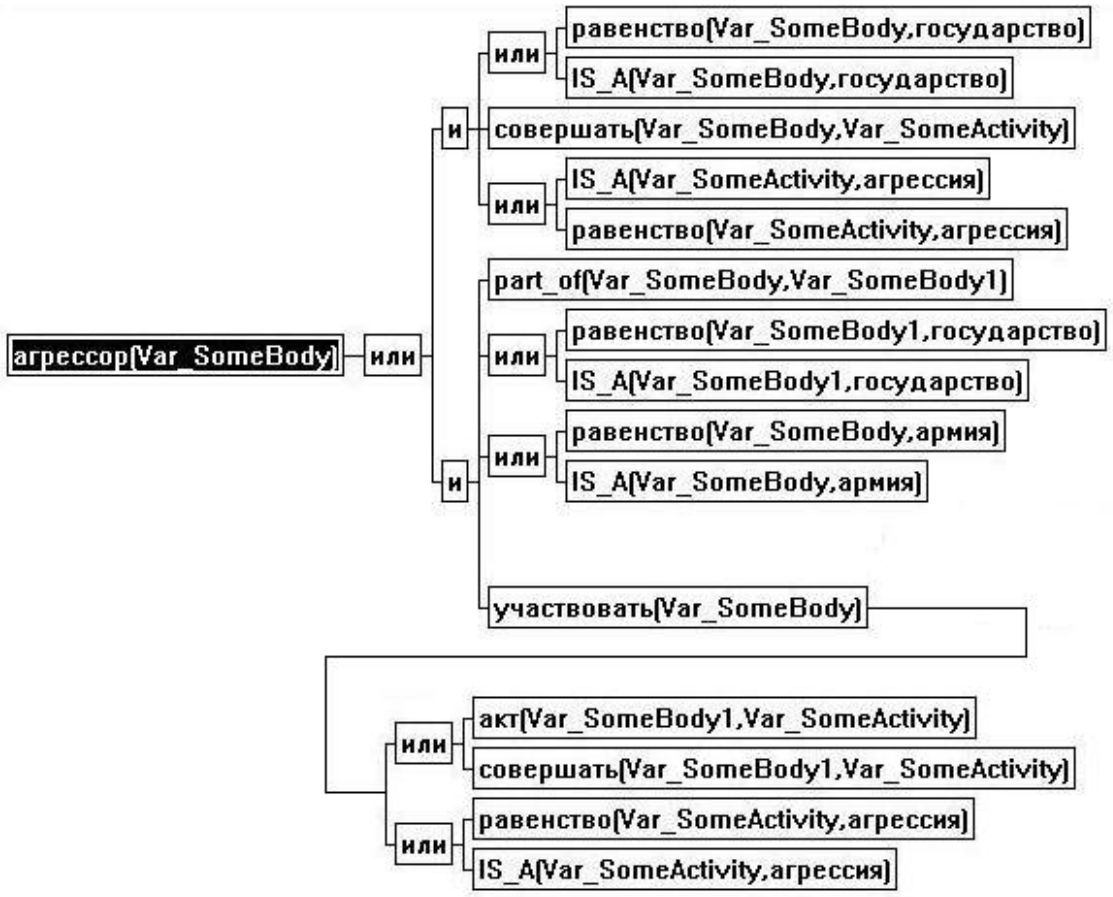


Рис. 3.5. Обобщенная теория ЛЗ *“агрессор”*

Помимо ТКС [162], исходные варианты толкования были взяты из Большой советской энциклопедии, тематического словаря “Война и мир” и словаря Брокгауза и Ефрона [132]. В настоящей главе (кроме раздела 3.5) для визуализации решеток диаграммами линий используется специализированный программный продукт ToscanaJ [185], реализующий методы АФП.

Как видно из приведённого примера, введение в рассмотрение области, которую образуют элементы толкования заданного лексического значения в решетке формальных понятий, позволяет различать случаи:

– использования разных ХФ с одним и тем же значением в независимых альтернативных вариантах теории ЛЗ (“ИЛИ”-обобщение). В формальном контексте на рис. 3.4 примерами являются пара ФП (“*Толкование2_агрессор*”, “*Толкование3_агрессор*”) и пара, образованная “*Толкованием1_агрессор*” и НОСП для пары (“*Толкование2_агрессор*”, “*Толкование3_агрессор*”);

– описания одного и того же элемента толкования ЛЗ, но посредством разных ХФ (обобщение посредством отношения “И”). В представленном на рис. 3.4 формальном контексте примером может послужить содержание ФП “*Толкование1_агрессор*”, а также содержание НОСП для пары (“*Толкование2_агрессор*”, “*Толкование3_агрессор*”).

При этом вычислительная сложность процесса обобщения теорий заданного ЛЗ зависит исключительно от мощности множества характеристических функций. Согласно определению смысла как интенсионала лексического значения число самих ХФ не зависит от числа обобщаемых теорий. В перспективе для утверждений, объединяемых посредством отношения “ИЛИ”, здесь появляется возможность задействовать статистические методы для выявления наиболее значимых признаков.

3.2. Прецеденты семантических отношений для ситуаций синонимии на основе стандартных лексических функций

При формировании прецедентов СЭ для ситуаций использования лексических функций-параметров *актуальна задача* выявления и обобщения смыслового отношения в рамках расщепленного значения. В настоящем разделе мы рассмотрим, каким образом данная задача может быть решена с привлечением информации ЛЗ, формализуемого посредством теорий вида (3.1).

Пусть $Rap(rule_j)$ – множество условий применимости правила $rule_j \in \Pi$, W_1 и W_2 – комплексы лексических единиц, заменяемых посредством π согласно постановке *задачи 1.2*, а $W = W_1 \cup W_2$.

Определение 3.1. Пара (W_1, W_2) соответствует *расщепленному значению (РЗ)* при обязательном выполнении следующих условий:

1. $\forall w_i \in W_1$ либо является значением некоторой лексической функции для ключевого слова C_0, определяющего ситуацию СЭ, либо есть само C_0.

2. $\exists w_k \in W_1$: $w_k = F(C_0)$ и F относится к классу лексических функций-параметров [62, с. 78].

3. $W_2 = \{w\}$, при этом w есть либо значение некоторой ЛФ-замены [62, с. 78] для данного C_0, либо есть само C_0. Комплекс W_2 соответствует нерасщепленному смысловому эквиваленту расщепленного значения, отождествляемого с W_1.

Заметим, что перераспределение смысла между лексемами, актуальное для формализации $\forall rap_l \in Rap(rule_j)$, характерно для ситуаций с ЛФ-параметрами. В общем случае формирование прецедента для ситуации СЭ на основе РЗ предполагает наряду с формализацией требований к смыслу слов в составе каждого W_j, $j \in \{1, 2\}$, выявление и обобщение смыслового отношения между произвольными w_i и w_m, входящими в W и отвечающими нижеперечисленным требованиям:

1. $w_i \neq w_m$.

2. w_i есть значение некоторой лексической функции-параметра для заданного C_0.

3. w_m есть либо значение некоторой лексической функции-замены для заданного C_0, либо $w_m = C_0$.

Пример. РЗ *“осуществлять эксперимент”*, где значением ЛФ $Oper_1$ задается смысловое отношение наподобие *“операция с”* между 1-м участником ситуации СЭ (кто осуществляет эксперимент) и ее названием (*“эксперимент”*). Данное РЗ имеет нерасщепленный эквивалент *“экспериментировать”*.

Таким образом, требования к заменяемым лексическим единицам, предъявляемые условием rap_l, определяются смысловыми отношениями между ключевым словом C_0 и его лексическими коррелятами, которые входят в заменяемый комплекс лексических единиц. В лексической семантике именно такие отношения и описываются стандартными лексическими функциями. Фактически для ситуации СЭ на основе расщепления лексического значения расщепленное значение опреде-

ляет этот вид отношений. Указанный факт позволяет поставить задачу выявления и обобщения смыслового отношения в рамках РЗ по аналогии с описанием семантики именных групп на основе формализованного представления толкований лексических значений слов в виде теорий (3.1).

Сказанное подтверждается наработками по Русскому общесемантическому словарю (РОСС): лексические функции используются в качестве семантических характеристик (СХ) отдельных слов в РОСС. А это означает, что такие слова могут быть и названиями отношений в утверждениях теорий других слов. Примером может послужить значение ЛФ $Oper_1$ для ЛЗ *“эксперимент”* (то есть *“осуществлять”*) (рис. 3.1), которое присутствует в одном из утверждений теории ЛЗ *“экспериментировать”*. При этом применение лексических функций в качестве СХ отдельных слов в указанном словаре позволяет сделать вывод о возможности выявления смысловых зависимостей, определяемых лексическими функциями, путем сравнительного анализа множеств аксиом теорий ЛЗ слов в расщепленном значении.

Утверждение 3.4. Смысловое отношение F, значимое для формирования $rap_l \in Rap(rule_j)$, между некоторым словом w_2 и его лексическим коррелятом w_1, входящим в РЗ, будет иметь место тогда, когда

$$LM_1 = LM_{11} \cup LM_{22} \cup LM_{12},$$

$$LM_2 = LM_{11} \cup \{(F, o, LM_{22})\} \cup LM_{12},$$

$$LM_{11} \cap LM_{22} = \varnothing,\ LM_{11} \cap LM_{12} = \varnothing,\ LM_{12} \cap LM_{22} = \varnothing,$$

где LM_1 – набор утверждений теории ЛЗ для w_1;

LM_2 – аналогичный набор для w_2;

тройка (F, o, LM_{22}) есть утверждение вида (3.2).

При независимом построении теорий для одного и того же слова (но разными исследователями и на основе разных корпусов текстов) возникает задача контроля адекватности и полноты сочетаемости слова по заданной ЛФ. В следующем разделе мы покажем, каким образом данная задача может быть решена

совместным использованием информации моделей управления предикатных слов и формализованных теорий лексических значений.

3.3. Семантика расщепленного значения и смысловые валентности предикатного слова

В работе [83] было рассмотрено использование семантической информации предикатных слов русского языка, представленной в Русском общесемантическом словаре, для безошибочной идентификации отношения “более общее – более частное” (в терминологии АФП – “подпонятие – суперпонятие”) между предикатными словами на основе анализа ролевого состава их ЛЗ. Следует отметить, что описание дифференциальных признаков слова цепочками СХ в указанном словаре есть разновидность формульного описания (3.1) для теории СК этого слова. Каждая СХ соответствует некоторой “семантической координате” (сорту) [139], обозначаемой словом сущности. К настоящему моменту идеология РОСС имеет практическое воплощение в разработанном рабочей группой Aot.ru автоматизированном рабочем месте (АРМ) лингвиста [2].

Использование лексических функций в качестве СХ отдельных слов в РОСС позволяет сделать вывод об использовании таких слов в качестве названий отношений в утверждениях теорий других слов, а следовательно, и возможности выявления смысловых зависимостей, определяемых лексическими функциями, путем сравнительного анализа множеств аксиом теорий ЛЗ слов в расщепленном значении. Согласно *утверждению 3.4* сравнение производится на предмет наличия зависимости, определяемой семантическим отношением в некотором постулате вида (3.2) или (3.3) одной из сопоставляемых теорий. При этом подмножество аксиом теории ЛЗ другого слова либо является одним из аргументов этого отношения, либо непосредственно задается одним из сравниваемых слов. Примером могут послужить теории ЛЗ “*эксперимент*” и “*экспериментировать*”, представленные на рис. 3.1.

Лексическими функциями описывается в первую очередь лексическая сочетаемость, которая определяется лексическим значением ключевого слова ЛФ-синонимической замены. Следовательно, ЛЗ более узкого по смыслу слова (в терминологии АФП – гипонима) включает лексические значения более широких по смыслу слов (гиперонимов), которые упоминаются в толковании ЛЗ рассматриваемого слова, а следовательно, и в его теории. Таким образом, слово-гипоним в большинстве случаев будет иметь в качестве значений ЛФ-параметра значения этой же ЛФ для тех слов-гиперонимов, которые упоминаются в его толковании (теории). Сказанное позволяет описать для заданной ЛФ систему слов, являющихся ее аргументами, посредством следующего формального контекста[5]:

$$Klf = (Glf, Mlf, Ilf), \tag{3.6}$$

где множество объектов Glf есть множество ключевых слов-аргументов заданной лексической функции. Признаковому множеству Mlf соответствует множество слов-значений лексической функции для слов из Glf. Бинарное отношение $Ilf \subseteq Glf \times Mlf$ задает частичное отображение Glf на Mlf и ставит в соответствие каждому ключевому слову $C_0 \in Glf$, определяющему ситуацию СЭ, множество значений заданной лексической функции.

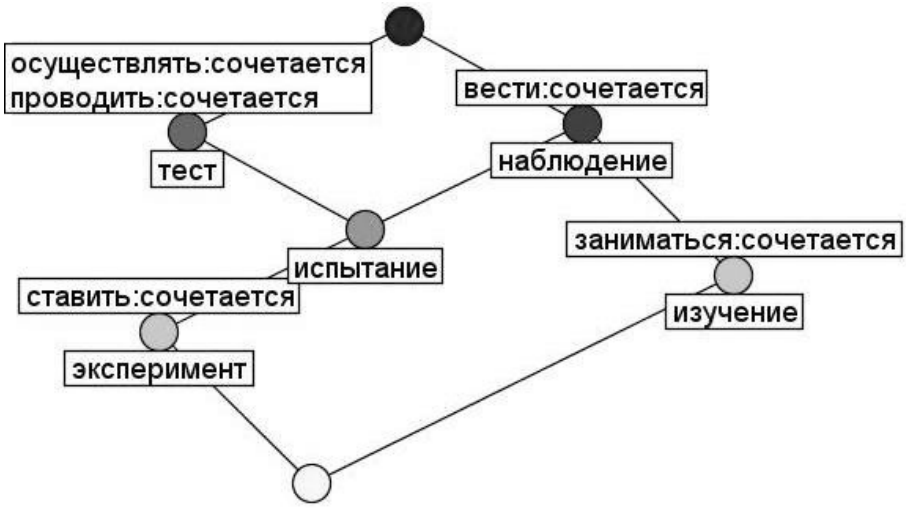


Рис. 3.6. Слова-аргументы лексической функции $Oper_1$ из верхней окрестности для лексического значения *“эксперимент”*

[5] В обозначениях из формулы (3.6) *“lf”* есть сокр. от англ. lexical function – лексическая функция.

В качестве примера на рис. 3.6 представлена решётка ФП контекста вида (3.6) для слов-аргументов ЛФ $Oper_1$ из верхней окрестности ЛЗ *“эксперимент”*.

С другой стороны, для предикатных слов отношение “гипоним – гипероним” определяется, как было показано в [83], в первую очередь анализом смысловых валентностей. Поэтому для оценки адекватности классификации объектов множества Glf на основе формального контекста (3.6) рассмотрим определение отношения гипонимии между семантическими классами с учетом формульных описаний вида (3.1) для семантических характеристик слова.

Пусть для каждого слова w_i мы имеем описание его семантического класса Csf_i посредством четверки[6]:

$$Ssf_i = \left(Csf_i, Lsf_i, Dsf_i, Dsf_{ij}\right), \tag{3.7}$$

где второй, третий и четвертый элементы указывают на дескрипторы, используемые в РОСС для однозначной идентификации Csf_i. При этом компонент Lsf_i есть список дескрипторов семантических характеристик в последовательности “более общая СХ – более специфическая СХ”. Дескрипторы Dsf_i и Dsf_{ij} обозначают таксономическую категорию и ее подкласс соответственно.

Предположим также, что w_i есть предикатное слово. При этом для его семантического класса имеется описание характеризованного ролевого состава[7]:

$$Cact = \left(Csf_i, Lr_i\right), \tag{3.8}$$

где $\forall Act_{ti} \in Lr_i$ включает название $Role_{ti}$ роли плюс список Lc_{ti} возможных семантических классов актанта:

$$Act_{ti} = \left(Role_{ti}, Lc_{ti}\right). \tag{3.9}$$

Утверждение 3.5. ЛЗ слова, относящегося к СК Csf_1:

$$Cact_1 = \left(Csf_1, Lr_1\right),$$

следует считать суперпонятием для ЛЗ слова СК Csf_2:

---

[6] Здесь *“sf”* есть сокр. от англ. semantic feature - семантическая характеристика.

[7] Здесь *“act”* есть сокр. от англ. actant - актант.

$$Cact_2 = (Csf_2, Lr_2),$$

тогда, когда для $\forall Role_{t2}$: $(Role_{t2}, Lc_{t2}) \in Lr_2$ $\exists (Role_{t1}, Lc_{t1}) \in Lr_1$, такой, что каждому $Scl_{at1} \in Lc_{t1}$ можно поставить в соответствие $Scl_{at2} \in Lc_{t2}$, который либо равен Scl_{at1}, либо связан с Scl_{at1} отношением “вид – род”.

Утверждение 3.6. ЛЗ слова w_i, относящегося к СК Csf_i:

$$Ssf_i = (Csf_i, Lsf_i, Dsf_i, Dsf_{ij}),$$

следует считать суперпонятием для ЛЗ слова w_m СК Csf_m:

$$Ssf_m = (Csf_m, Lsf_m, Dsf_i, Dsf_{ij}),$$

если в дополнение к определенным *утверждением 3.5* условиям при отсутствии для актанта $Act_{ai} = (Role_{ai}, Lc_{ai}) : Act_{ai} \in Lr_i$ актанта подпонятия с показанным в *утверждении 3.5* соответствием набора возможных СК существует актант $Act_{bm} = (Role_{bm}, Lc_{bm}) : Act_{bm} \in Lr_m$, отвечающий нижеследующему требованию. Пусть для $\forall Csf_{qai} \in Lc_{ai}$ задано описание

$$Ssf_{qai} = (Csf_{qai}, Lsf_{qai}, Dsf_{qai}, Dsf_{qai1})$$

и аналогично для $\forall Csf_{sbm} \in Lc_{bm}$

$$Ssf_{sbm} = (Csf_{sbm}, Lsf_{sbm}, Dsf_{sbm}, Dsf_{sbm2}).$$

Тогда наряду с вхождением в список Lsf_{sbm} семантических характеристик из списка Lsf_{qai} некоторым семантическим характеристикам $SF_{pqai} \in Lsf_{qai}$ ставятся в соответствие формализованные описания (3.1):

$$Lm(SF_{pqai}) = (SF_{pqai}, LM_{pqai}),$$

причем $\exists Lsf'_{sbm} \subset Lsf_{sbm}$: $\forall SF_{osbm} \in Lsf'_{sbm}$ является в составе LM_{pqai} либо одним из элементов тройки (3.2), либо первым элементом тройки (3.3).

Примером указанного соответствия может послужить аспектная валентность у ЛЗ “*испытание*” и валентность содержания у ЛЗ “*тест*” из представленных на рис. 3.6 слов верхней окрестности ЛЗ “*эксперимент*”.

Действительно, согласно указанному в *утверждении 3.5* условию существования отношения гипонимии между лексическими значениями ЛЗ *“тест”* не может выступать в качестве суперпонятия для ЛЗ *“испытание”*. Основание – отсутствие задаваемого *утверждением 3.5* соответствия для валентности аспекта у ЛЗ *“испытание”* и валентности содержания у ЛЗ *“тест”*. Тем не менее в словарной базе данных АРМ лингвиста [2] для семантического класса слова, реализующего аспектную валентность у ЛЗ *“испытание”*, и для семантического класса слова, реализующего валентность содержания у ЛЗ *“тест”*, представлены описания совокупностями вышеупомянутых дескрипторов семантических характеристик, таксономических категорий и их подклассов.

Имеем:

w_i = “*тест*”, w_m = “*испытание*”,

Ssf_{qai} = (“*ситуация*”, [“SITUAT”], “LABL”, “SIT”),

Ssf_{sbm} = (“*свойство*”, [“ATTR”], “ASP”, “*Не определена*”).

Кроме того, имеем также теорию сорта, отождествляемого с СХ “SITUAT” (рис. 3.7).

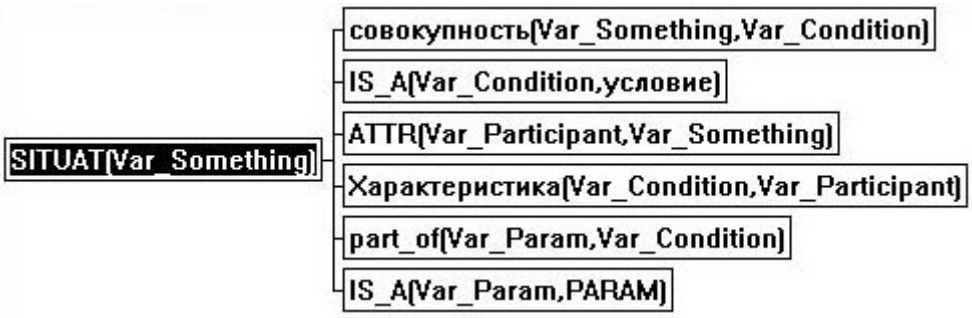


Рис. 3.7. Теория сорта “SITUAT”

Как видно из приведенного на рис. 3.7 древовидного описания, теория сорта “SITUAT”, упоминаемого в списке СХ для ЛЗ *“ситуация”*, “ссылается” на семантические характеристики “ATTR” и “PARAM”, из которых “ATTR” присутствует в списке СХ для ЛЗ *“свойство”*. Таким образом, относительно ЛЗ *“ис-*

пытание” ЛЗ *“тест”* удовлетворяет сформулированным нами требованиям к суперпонятию лексического значения.

Визуализируя (рис. 3.8) средствами Visual Prolog'a отношение гипонимии для множества семантических классов слов-аргументов заданной ЛФ, мы можем оценить как адекватность и полноту описания слова по ЛФ, так и корректность лексикографического толкования как основы для построения модели управления этого слова (рис. 3.9).

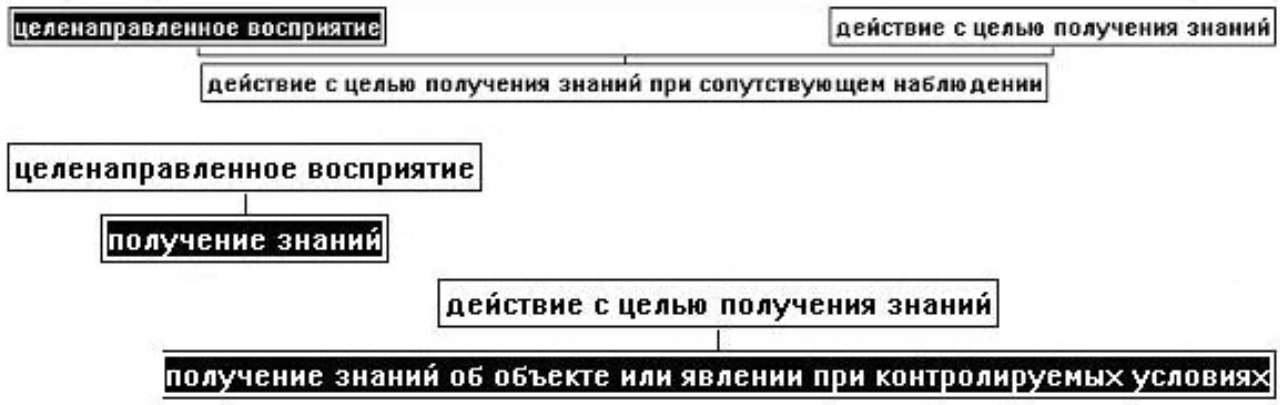


Рис. 3.8. Семантические классы слов окрестности ЛЗ *“эксперимент”*

Таблица 3.1

Слова окрестности ЛЗ “*эксперимент*” и их семантические классы

Слово	Семантический класс
Эксперимент	Получение знаний об объекте или явлении при контроли-руемых условиях
Испытание	Действие с целью получения знаний при сопутствующем наблюдении
Изучение	Получение знаний
Тест	Действие с целью получения знаний
Наблюдение	Целенаправленное восприятие

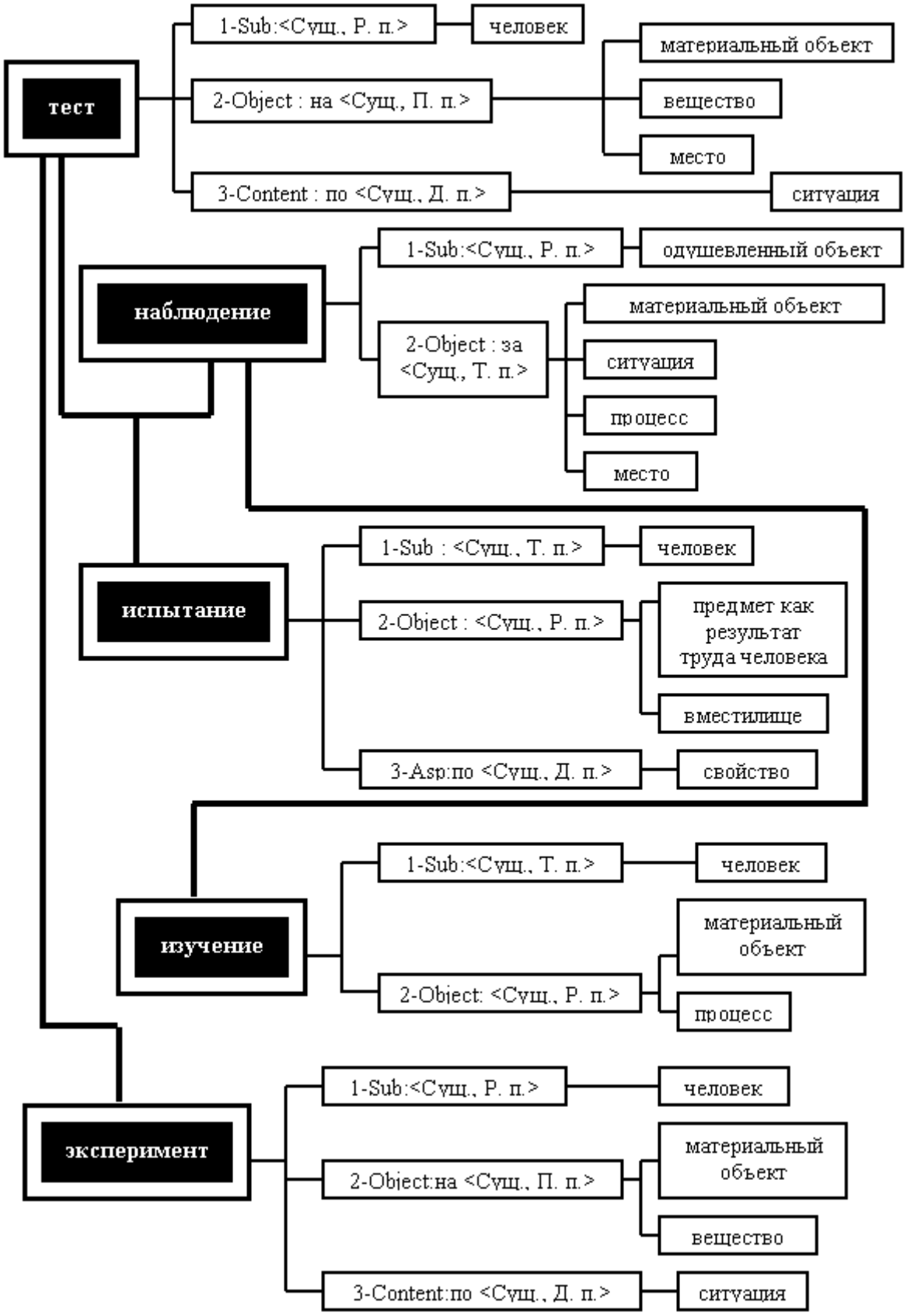


Рис. 3.9. Ролевой состав слов окрестности ЛЗ *“эксперимент”*

Замечание. Фактически *утверждением 3.6* определяется отношение порядка на множестве предикатных слов для случая зависимости между их семантическими характеристиками. При этом взаимно-однозначное соответствие между семантическими классами актанта гипонима и гиперонима устанавливается путем поиска общих подсписков семантических характеристик в совокупности с вхождением семантических характеристик одного актанта в утверждения теорий для семантических характеристик другого актанта.

Пусть W_1 и W_2 – комплексы лексических единиц, заменяемых посредством некоторого правила $rule_j \in \Pi$ согласно постановке *задачи 1.2*, W_1 отождествляется с РЗ, а W_2 – с нерасщепленным смысловым эквивалентом этого РЗ. Положим также, что заданы формульные описания $Lm(w_1)$ и $Lm(w_2)$ вида (3.1) для ЛЗ слов $w_1 \in W_1$ и $w_2 \in W_2$ соответственно. Обозначим множество, каждый элемент которого входит либо в W_1, либо в W_2 и является предикатным словом, как Ws. При этом для каждого $w_i \in Ws$ имеется описание характеризованного ролевого состава посредством двойки (3.8).

Утверждение 3.7. Будем считать, что $Lm(w_1)$ и $Lm(w_2)$, $\{w_1, w_2\} \subset Ws$, адекватно задают $rap_l \in Rap(rule_j)$ при выполнении следующих условий:

1. На множестве Ws может быть определено отношение порядка ($\leq$) в соответствии с условиями в *утверждениях 3.5* и *3.6*;

2. Между w_2 и w_1 существует смысловое отношение F в соответствии с условиями, задаваемыми *утверждением 3.4*;

3. Само имя отношения F в составе формального контекста (3.6) принадлежит множеству формальных признаков ЛЗ слова w_{Sup}, составляющего объем формального понятия, не превышающего наименьшего общего суперпонятия для множества Nh тех формальных понятий, объемы которых включают слова верхней окрестности лексического значения w_1. Формально $Nh \subset \Re(Gh, Mh, Vh, Ih)$, при этом $Gh \supset Ws$, а Mh есть множество возможных

ролевых ориентаций актантов (3.9) для ситуаций, обозначаемых предикатными словами $w_m \in Gh$. Множество Vh есть множество всех множеств семантических классов слов, способных замещать некоторую валентность $Role_{ti}$ предикатного слова $w_m \in Gh$, а $Ih \subseteq Gh \times Mh \times Vh$.

Требования к РЗ, в состав которого входит слово w_{Sup}, определяются аналогично.

3.4. Экспериментальная апробация методики формирования прецедентов смысловой эквивалентности на материале тезауруса по анализу изображений

Разработанная методика формирования прецедентов для классов СЭ, определяемых на основе расщепленных значений с лексическими функциями-параметрами, была апробирована на материале специализированного тезауруса по анализу изображений, предложенного и развиваемого исследовательским коллективом Вычислительного центра им. А.А. Дородницына Российской академии наук. Концепции такого тезауруса и ее техническому воплощению был посвящен ряд публикаций наших коллег, в частности [135–138, 142, 155].

Следует отметить, что формализация знаний в области обработки, анализа и понимания изображений является неотъемлемой составляющей построения интеллектуальных систем, способных выполнять функцию партнера человека при обработке больших массивов разнотипной информации, поступающей независимо из различных источников. Первым шагом на пути к созданию таких систем является построение онтологии той предметной области, которая включает обработку, анализ и распознавание изображений. При этом логико-понятийную основу онтологии составляет тезаурус, основным требованием к которому является динамичность. Тезаурус интеллектуальной системы должен быть не только средством представления современного состояния рассматриваемой области знания, должен не только

включать все основные понятия и фиксировать существующие связи между этими понятиями, но и быть гибким инструментом интеграции новых и уже имеющихся знаний, обобщения и систематизации знаний, отслеживания противоречий в той информации, которая заносится в тезаурус.

Приведенный далее, на рис. 3.10–3.17, пример показывает, каким образом предложенный в настоящей главе подход к описанию смысла слова набором характеристических функций позволяет решить указанные задачи, возлагаемые на тезаурус, а также уменьшить объем памяти ЭВМ, занимаемый самим тезаурусом.

Рис. 3.10. Вариант 1 теории ЛЗ *“изображение”*

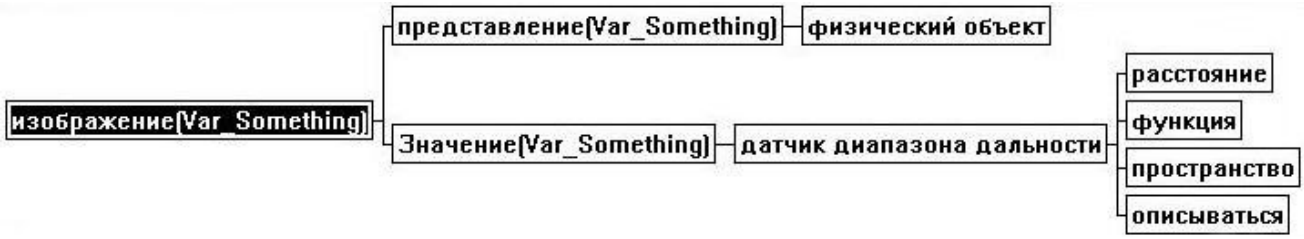


Рис. 3.11. Характеристические функции и формальные признаки их значений – вариант 1

Рис. 3.12. Вариант 2 теории ЛЗ *“изображение”*

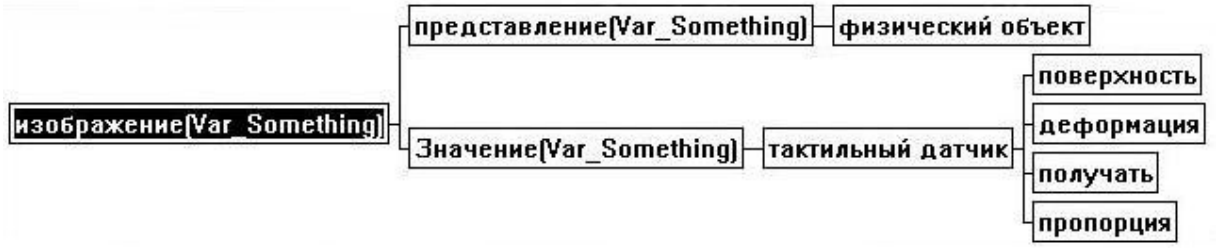


Рис. 3.13. Характеристические функции и формальные признаки их значений – вариант 2

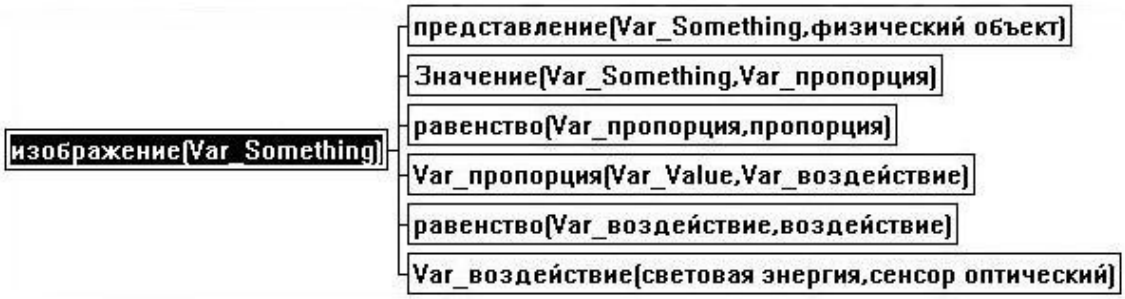


Рис. 3.14. Вариант 3 теории ЛЗ *“изображение”*

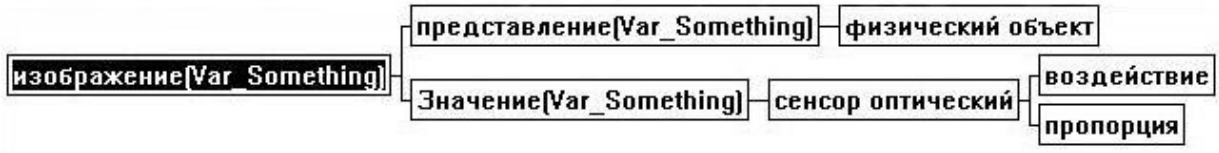


Рис. 3.15. Характеристические функции и формальные признаки их значений – вариант 3

При этом для обобщения независимых вариантов толкования лексического значения слова используются математические методы АФП, хорошо зарекомендовавшие себя в лингвистических приложениях [179], и реализующее эти методы программное обеспечение, свободно распространяемое в сети Internet. Это дает возможность распараллелить работу по созданию тезауруса заданной предметной области между исследовательскими коллективами разных научных школ, а посредством концептуальной кластеризации сопоставлять различные точки зрения на тот или иной термин (понятие).

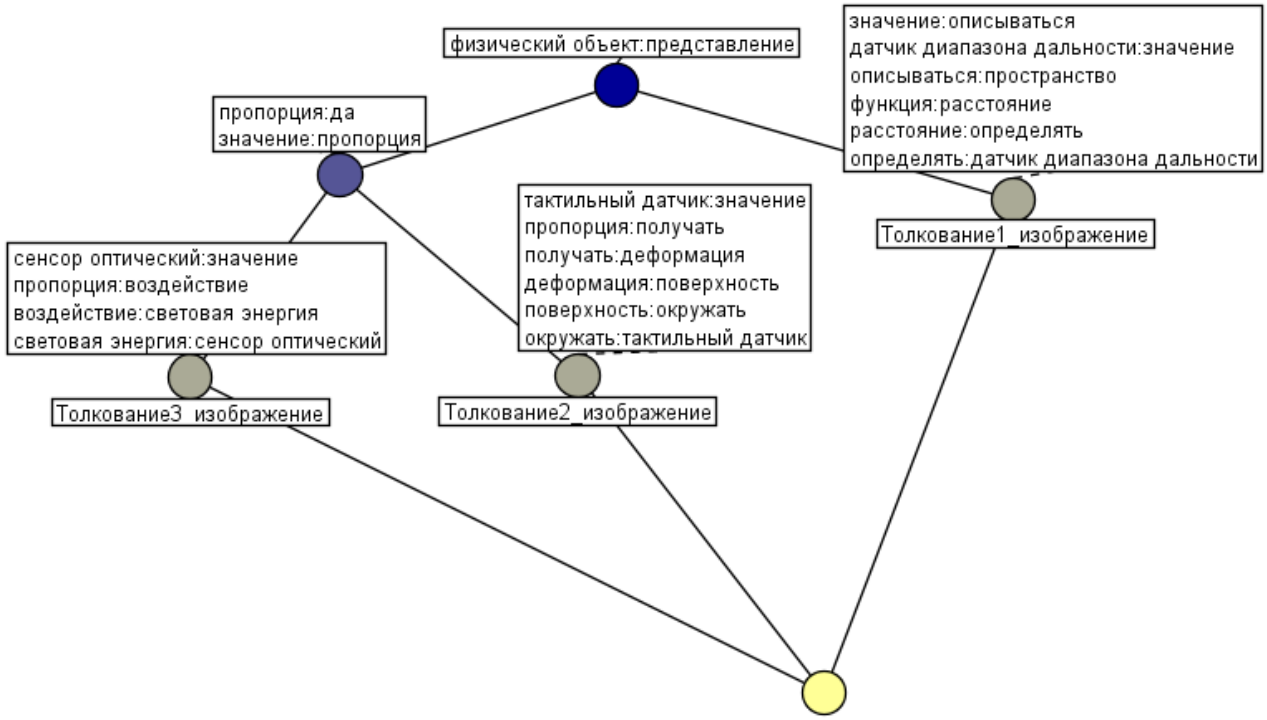


Рис. 3.16. Решетка формальных понятий для независимых толкований ЛЗ *“изображение”*

Задействование характеристических функций при описании смысла слова и их выводимость из теории его лексического значения позволяет в перспективе ввести в рассмотрение родовидовые зависимости между теориями на основе решеток, получаемых по нескольким независимым вариантам толкования одного и того же лексического значения (рис. 3.16). При этом базис импликаций [153] формального кон-

текста (3.5) может послужить основой изучения взаимозаменяемости элементов толкования относительно различных характеристических функций.

Тем не менее следует отметить, что основой информационного наполнения рассматриваемого тезауруса являются тематические публикации по заданной предметной области.

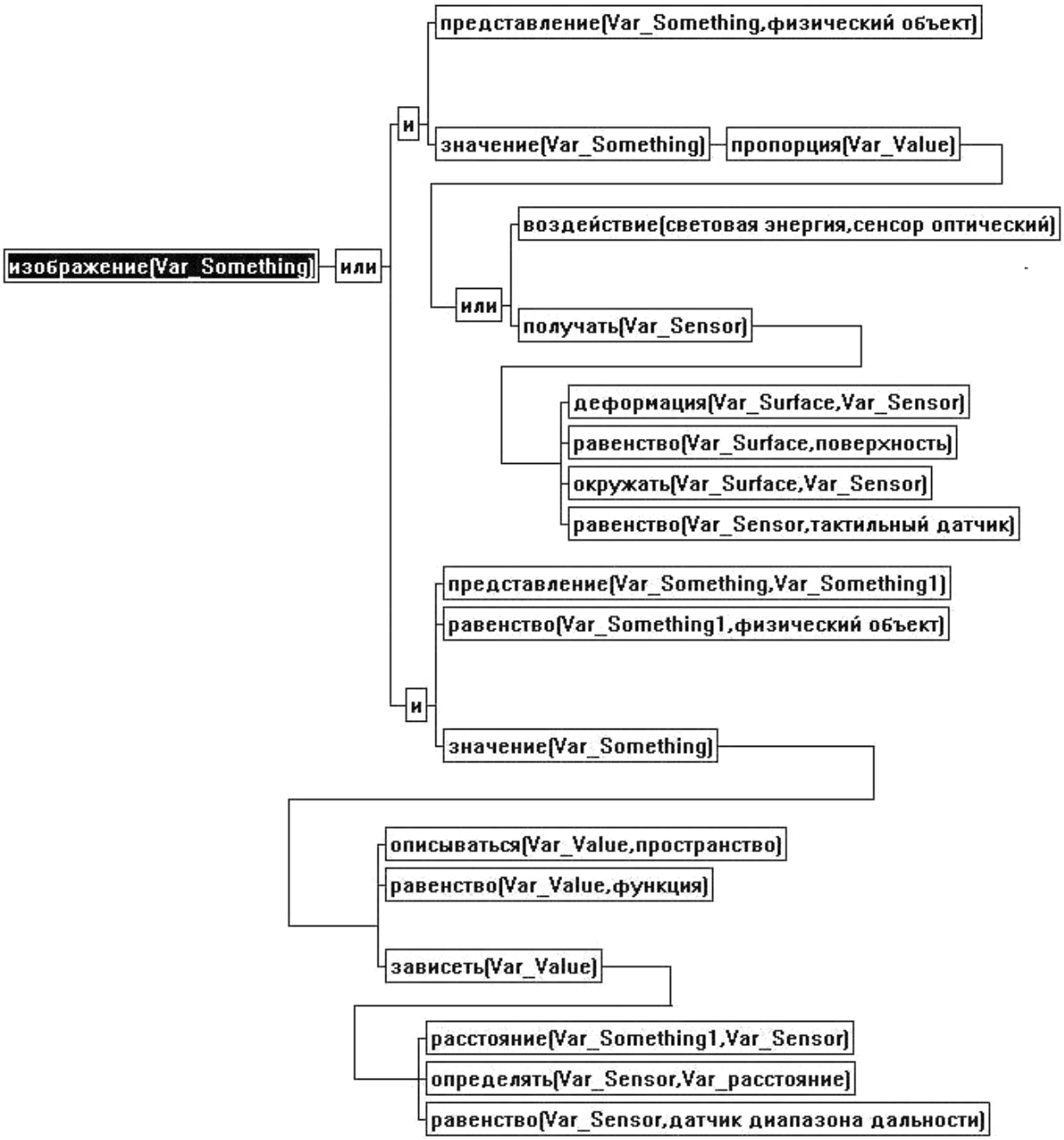


Рис. 3.17. Обобщение утверждений независимых теорий для ЛЗ *“изображение”*

На практике сказанное означает необходимость не только систематизации уже накопленных знаний, но и автоматизированного получения новых непосредственно

из текстов (научных статей, тезисов докладов, монографий), формируемых носителем предметных знаний – человеком. В частности, для генерации формализованных описаний вида (3.1) требуется решение задачи формирования и кластеризации отношений, на основе которых строятся утверждения теорий. Этому вопросу посвящен следующий раздел.

3.5. Формирование отношений в естественном языке на основе множеств семантически эквивалентных фраз

Как было показано в главе 1, языковой опыт человека можно разделить в соответствии с разделением концептуальной картины мира. При этом основополагающим является понятие ситуации употребления ЕЯ, представляемой посредством тройки (1.1) и рассматриваемой как основа языкового генезиса. Предположим теперь, что в качестве элементов множества Ts в составе тройки (1.1) выступают синонимичные (с точки зрения носителя языка) ЕЯ-фразы, причем каждая из них описывает одну ситуацию действительности (относительно языкового контекста ситуации S). Положим выбор ЕЯ-фраз $Ts_i \in Ts$ для описания S равновероятным.

Поскольку S есть (по определению) полное и независимое описание языкового контекста, то имеем следующую задачу.

Задача 3.1. На основе ЕЯ-фраз множества Ts сформировать отношения, представляемые множеством R в модели (1.1), рассматривая отношения между объектами $o \in O$ в качестве признаков последних относительно ситуации S .

Рассмотрим текст $Ts_i \in Ts$ с точки зрения символов, которые его составляют. У каждого текста Ts_i выделяется некоторая неизменная часть Tc_i, общая для всех $Ts_i \in Ts$, и флективная часть Tf_i. На множестве Tf_i выражаются синтагматические зависимости, которые задаются с помощью синтаксических отношений и определяют возможность сосуществования словоформ в линейном ряду. Аналогично для слова w_{ij} имеем:

$$W_{ij} = Wc_{ij} \bullet Wf_{ij}, \qquad (3.10)$$

где W_{ij} – буквенный состав слова; $Wc_{ij} \subset Tc_i$ составляют символы неизменной части слова, именуемой далее основой; $Wf_{ij} \subset Tf_i$ – символы флективной части, именуемой далее флексией; символом "$\bullet$" обозначается конкатенация символьных последовательностей.

Таким образом, попарным сравнением W_{ij} различных Ts_i требуется найти:

1) Wc_{ij} и Wf_{ij} каждого W_{ij} при $\left|Wc_{ij}\right| \to \max$;

2) отношение R_q, определяющее допустимость сочетания $\left(Wf_{ij}, Wf_{ik}\right)$, $k \neq j$.

Введем в рассмотрение индексное множество J для неизменных частей всех слов, употребленных во всех фразах из $Ts_i \in Ts$.

Определение 3.2. Упорядоченную совокупность индексов $j \in J$ неизменных частей слов, присутствующих в $Ts_i \in Ts$, будем называть *моделью линейной структуры* этой фразы, $Ls\left(Ts_i\right)$.

При этом порядок индексов в $Ls\left(Ts_i\right)$ идентичен порядку следования соответствующих слов в Ts_i. Поэтому $Ls\left(Ts_i\right)$ позволяет однозначно восстановить ЕЯ-фразу Ts_i на множестве всех слов для всех $Ts_i \in Ts$. И, наоборот, для $\forall Ts_i \in Ts$ на индексном множестве J можно однозначно построить $Ls\left(Ts_i\right)$.

Для построения множества R в составе тройки (1.1) необходимо найти совокупность указанных моделей, удовлетворяющих требованиям проективности. С учетом линейной природы синтагм дополним ограничения на проективность, рассмотренные, в частности, в [45] и используемые в системах анализа текстов, следующим образом.

Пусть $h\left(j, Ls\left(Ts_i\right)\right)$ – позиция индекса j в модели $Ls\left(Ts_i\right)$. Тогда множество связей относительно $Ls\left(Ts_i\right)$ можно определить как

$$D: Ts_i \to \{(h(j, Ls(Ts_i)), h(k, Ls(Ts_i))) : j \neq k\}.$$

Определение 3.3. *Связь* $d_{qi} = (h(j, Ls(Ts_i)),\ h(k, Ls(Ts_i)))$ является *допустимой для модели* $Ls(Ts_i)$, если $\exists\{Ts_l, Ts_m\} \subset Ts$, $l \neq m$, причем и $Ls(Ts_l)$, и $Ls(Ts_m)$ содержат в качестве подпоследовательности либо $\{j,k\}$, либо $\{k, j\}$. При этом пара индексов (j,k) соответствует одной синтагме, а индекс q – типу синтаксического отношения, которое ей соответствует.

Положим, что для $\forall Ts_i \in Ts$, $i = 1,\ldots,|Ts|$, все $d_{qi} \in D(Ts_i)$ удовлетворяют *определению 3.3*.

Определение 3.4. Будем считать, что модель $Ls(Ts_i)$ проективна относительно множества R в составе тройки вида (1.1), если $\sum_{q=1}^{|D(Ts_i)|} \Delta_{qi} \leq |Ls(Ts_i)|$, где $\Delta_{qi} = |h(j, Ls(Ts_i)) - h(k, Ls(Ts_i))|$.

На основе $\bigcup_i D(Ts_i)$ формируется граф синтагм (V_J, I_J). Элементами множества вершин V_J этого графа являются множества пар (j,k), $\{j,k\} \subset J$, сгруппированных по некоторому общему для них индексу k. Множества E_1 и E_2, входящие в V_J, будут соединены ребром из I_J, если $\exists\{j,k,m\} \subset J$: $(j,k) \in E_1$, $(k,m) \in E_2$ и $j \neq m$.

Анализом (V_J, I_J) строится дерево синтаксических связей (V_{JT}, I_{JT}).

Формально

$$V_{JT} = J,\ I_{JT} = \{(j,k) : \exists E \in V_{JT}, (j,k) \in E\}. \qquad (3.11)$$

При этом индекс $k \in V_{JT}$ соответствует корню дерева (3.11), если $\exists E_1 \in V_J$, в котором пары индексов сгруппированы по k, $|E_1| > 1$, а k не содержится ни в одной паре индексов для $\forall E_2 \in V_J$: $E_1 \neq E_2$.

Содержательно корень соответствует предикатному слову (глаголу либо отглагольному существительному), которое (по определению) обозначает ситуацию. Согласно данному в главе 1 определению семантического отношения наибольший

интерес для *задачи 3.1* представляют ситуации вида (1.1) с двумя и более участниками, поэтому число дочерних узлов у корня полагается больше одного.

Будем использовать маршруты в дереве (3.11) для выделения классов отношений множества R в модели (1.1) согласно сформулированной нами *задаче 3.1*. Данная задача наиболее естественно решается методами АФП.

Рассмотрим множество флексий как множество формальных объектов $Gfl = \{ f_{ij} : f_{ij} = \bullet(Wf_{ij}) \}$, где $i = 1,\ldots,|Ts|$, а символом "$\bullet$" обозначается операция конкатенации, которая последовательно выполняется над символами из Wf_{ij}.

Введем в рассмотрение формальный контекст:

$$Kfl = (Gfl, Mfl, Ifl), \tag{3.12}$$

в котором $Mfl = Gfl$, а $Ifl \subseteq Gfl \times Mfl$. При этом

$$Ifl = \{ (f_{ij}, f_{ik}) : s(j,k) = true, \{j,k\} \subset J \}.$$

Отношение s определяется рекурсивно на основе (V_J, I_J):

1) $s(j_1, j_1) = true$;

2) $s(j_1, j_2) = true$ в одном из следующих двух случаев:

   – $\exists E_1 \in V_J : (j_1, j_2) \in E_1$, причем $\exists\, j_3 \in J$, для которого $s(j_2, j_3) = true$;

   – $\exists (E_1, E_2) \in I_J$: $\exists\, j_3 \in J$, при этом $(j_1, j_3) \in E_1$, $(j_3, j_2) \in E_2$, а $s(j_3, j_2) = true$.

Модель (3.12) выделяет классы отношений в R по характеру изменения флективной части зависимого слова.

Рассмотрим задачу поиска флексий для слов в составе расщепленных значений, семантику которых мы обсуждали в разделе 3.3. Здесь мы рассмотрим общий случай расщепленного предикатного значения (РПЗ) как совокупности вспомогательного глагола (связки) и некоторого существительного, называющего ситуацию. Для слов в составе РПЗ, как и для конверсивов (слов, обозначающих ситуацию с точки зрения разных ее участников), представления вида (3.10) не могут быть найдены попарным сравнением буквенного состава слов во всех $Ts_i \in Ts$.

Рассмотрим $Tcnc_i = \{w_{ij} : w_{ij} = \bullet(W_{ij})\}$, где символом "$\bullet$" обозначается операция конкатенации, последовательно выполняемая над символами из Wf_{ij}. Положим также, что последовательность $Tp_i \subset Ts_i$ определяет последовательность $Pcnc_i = \{u_k : u_k = \bullet(Wp_k), \bigcup_k Wp_k = Tp_i\}$, где $Wp_k \in Ts_i$ – последовательность символов слова, для которого не выделены неизменная и флективная часть.

Теорема 3.1. Последовательность $Pcnc_i$ содержит предикатное слово, если $\exists\{j,0,k\} \subset Ls(Ts_i)$: $\{w_{ij}, u_1, \ldots, u_p, w_{ik}\} \subset Tcnc_i$, где $\{u_1, \ldots, u_p\} = Pcnc_i$, $p = |Pcnc_i|$.

Доказательство следует из определения корня дерева (V_{JT}, I_{JT}) и проективности $Ls(Ts_i)$.

Пусть для последовательности $Pcnc_i$ выполняется условие *теоремы 3.1*.

Теорема 3.2. Слово $u_k \in Pcnc_i$ принадлежит расщеплённому предикатному значению, если $\exists Ts_j \in Ts$: $Ls(Ts_j) \neq Ls(Ts_i)$, а $u_k \in Pcnc_j$, причём $Pcnc_j$ также отвечает условию *теоремы 3.1*. При этом $\neg\exists Ts_k \in Ts$, где $Pcnc_k \subset Pcnc_i$ и отвечает *теореме 3.1*, а $Ls(Ts_k) \neq Ls(Ts_j)$ и $Ls(Ts_k) \neq Ls(Ts_i)$.

Доказательство следует из доказанной *теоремы 3.1* и определения множества ребер в графе (V_J, I_J).

Замечание. При выполнении условия *теоремы 3.2* u_k может быть в том числе и зависимым словом в составе РПЗ.

Пусть $Pcnc'_i$ – последовательность слов, удовлетворяющих *теореме 3.2*, а $Ts' \subset Ts$, при этом $Ts' = \{Ts_i : |Pcnc'_i| \to \max\}$.

Теорема 3.3. Для формирования тройки (3.12) при наличии РПЗ либо конверсива необходимо и достаточно найти множество $Ts' \subset Ts$: $Ts' = \{Ts_i : |Pcnc'_i| \to \max\}$.

Доказательство очевидным образом следует из доказанной *теоремы 3.2*.

Помимо выполнения условия *теоремы 3.3*, ключевым требованием при отборе $Ts_i \in Ts$ является минимум слов, не представимых в виде конкатена-

ции (3.10). Для $\forall u_k \in \bigcup_i Pcnc'_i$, $Ts_i \in Ts'$, его неизменная и флективная часть выделяются сравнением последовательности Wp_k его символов с аналогичными последовательностями Wp_j для всех $u_j \in \bigcup_l Pcnc_l : Ts_l \in (Ts \setminus Ts')$, а $Pcnc_l$ отвечает условию *теоремы 3.1*. При этом необходимо, чтобы $2|Wc_k| > |Wf_k| + |Wf_j|$, где $Wp_k = Wc_k \bullet Wf_k$, а $Wp_j = Wc_k \bullet Wf_j$.

Замечание. Если $Pcnc'_i \cap Pcnc_i \neq \varnothing$, то $\forall u_m \in (Pcnc_i \setminus Pcnc'_i)$ представляется вместе со словом слева от него в $Pcnc_i$ (в этом случае u_m рассматривается как предлог).

С учетом $Pcnc'_i$ дерево (3.11) преобразуется следующим образом:

1) корень изменяется с $k = 0$ на значение k для $u_k \in Pcnc'_i$ с максимальной встречаемостью в разных $Tcnc_i$ относительно заданной СЯУ;

2) левое поддерево остается без изменений;

3) правое поддерево перевешивается на узел j для $u_j \in Pcnc'_i$ наименьшей встречаемости;

4) в паре $\{u_l, u_m\} \subset Pcnc'_i$ дочерним будет узел для слова с меньшей встречаемостью.

В итоге основу формирования контекста (3.12) и множества R в составе тройки (1.1) составляют те Ts_i, которые наиболее полно представляют языковой контекст заданной ситуации.

Определение 3.5. Дерево (3.11), преобразованное согласно указанным правилам, будем далее называть расширенным деревом (3.11).

Заметим, что расширенное дерево (3.11) является деревом-прецедентом для множества $\{Tr_i : Ts_i = Synt(Tr_i)\}$ из определения компонента Ts в (1.1).

В заключении данного раздела рассмотрим свойства формального контекста (3.12), актуальные для выделения морфологических классов слов из фраз множества Ts', сформированного в соответствии с *теоремой 3.3*.

Пусть ℓ – базис импликаций, а $\Re fl$ – решетка формальных понятий для формального контекста Kfl.

Утверждение 3.8. ФП (Afl, Bfl): $Afl \subseteq Gfl$, $Bfl \subseteq Mfl$ соответствует предикатному слову, если $\exists(\Pr \to Cs) \in \ell$: $|\Pr| = 1$ и $\Pr \cup Cs = Bfl$. При этом наличие импликации $(\Pr_1 \to Cs_1) \in \ell : \Pr \subset Cs_1$ допускается только тогда, когда $\Pr_1 \cup Cs_1 = Bfl$.

Утверждение 3.9. ФП (Afl, Bfl): $Afl \subseteq Gfl$, $Bfl \subseteq Mfl$ соответствует слову (прилагательному либо причастию не в составе оборота), выполняющему в ЕЯ-фразе функцию определения, если Bfl есть множество признаков некоторого элемента множества Gfl и $\neg\exists(\Pr \to Cs) \in \ell$: $\Pr \cup Cs = Bfl$. Элементами Bfl при этом должны быть непустые строки. Если же множество Bfl состоит из единственного элемента – пустой строки, то данное ФП соответствует слову с синтаксической функцией наречия.

В противном случае ФП (Afl, Bfl) соответствует слову, выполняющему синтаксическую функцию существительного.

Синтаксические отношения как подмножество множества R в модели (1.1), выделяются анализом наименьшей верхней грани каждой пары ФП в $\Re fl$ и образуют классы по сходству характера флексии зависимого слова. Отдельному классу соответствует область в решетке, а наименьшая верхняя грань множества формальных понятий этой области – прецеденту класса. Следует отметить, что в настоящем разделе мы ведем рассмотрение только синтагматических зависимостей. Более широкие классы отношений, определяемые сочетанием основ главного и зависимого слов, а также сочетанием основ и флексий, выделяются аналогично. О формировании этих отношений пойдет речь в следующей главе работы.

Изложенный в настоящем разделе принцип формирования и кластеризации синтаксических отношений и его программная реализация, представленная в **приложении 1** диссертации фрагментами исходного текста на языке Visual Prolog 5.2, позволяет выделять произвольные отношения в рамках СЯУ за время, оце-

ниваемое сверху как квадрат произведения числа СЭ-фраз, определяющих СЯУ, и максимального числа слов во фразе.

Действительно, пусть СЯУ определяется тройкой (1.1), $m = |Ts|$, а n есть максимальная длина фразы из определяющих СЯУ.

При $m = 2$ имеем максимальное число сравнений, равное $n + n - 1 = 2n - 1$ – для первой итерации (поиск первой из пар слов, максимально близких по буквенному составу), $(n-1)+(n-1)-1 = 2n-3$ – для второй итерации (поиск второй из пар слов, максимально близких по буквенному составу), $(n-2)+(n-2)-1 = 2n-5$ – для третьей итерации, общее же число сравнений есть сумма n первых членов убывающей арифметической прогрессии и равняется $\frac{2(2n-1)+(n-1)(-2)}{2} n = n^2$. Очевидно, что для произвольного m поиск первой из возможных пар слов, наиболее близких по буквенному составу, требует максимум $mn + mn - 1 = 2mn - 1$ сравнение, поиск слова, наиболее близкого прецеденту сформированного таким образом класса – уже $(m-2)n + (m-2)n - 1 = 2n(m-2) - 1$ сравнение (с учётом того, что из каждой фразы в класс может войти максимум одно слово). В итоге число сравнений при формировании первого из классов слов по общности основы есть сумма m первых членов арифметической прогрессии, в которой:

– разность равна $(2n(m-2)-1)-(2mn-1) = -4n$;

– член с номером m равен $2mn - 1 + (m-1)(-4n) = 4n - 2mn - 1$.

Общее число сравнений, необходимое для выделения отдельного класса, оценивается сверху как $\frac{(2mn-1)+(4n-2mn-1)}{2} m = (2n-1)m$. Задача формирования следующего класса отличается числом слов во фразе, меньшим на единицу, поэтому оцениваемое число сравнений здесь $(2(n-1)-1)m = (2n-3)m$. Далее следует задача с оценкой числа сравнений, равной $(2(n-2)-1)m = (2n-5)m$. В итоге имеем арифметическую прогрессию с разностью $-2m$, при этом общее число сравнений для формирования всех классов оценивается как $\frac{(2n-1)m+m}{2} n = mn^2$.

Наихудший случай – отсутствие в множестве Ts фраз, имеющих в своём составе слова, для которых возможно выделение неизменной и флективной части сравнением буквенного состава с буквенным составом других слов. В этом случае каждый класс содержит ровно одно слово. При максимальной длине фразы, равной n, формирование классов слов по буквенному составу для первой из СЭ-фраз потребует $(m-1)n^2$ попарных сравнений слов, для второй – соответственно, $(m-2)n^2$, общее число сравнений вычисляется как сумма m первых членов убывающей арифметической прогрессии, для которой разность равна $-n^2$, член с номером m равен 0. В итоге общее число сравнений для формирования всех классов составляет $\frac{m(m-1)n^2}{2}$. Заметим, что такой случай представляет наименьший практический интерес при формировании отношений из множества R в (1.1), поскольку найти связи слов в соответствии с *определением 3.3* здесь не представляется возможным.

Следующий шаг после выделения классов слов по буквенному составу есть выделение связей, отвечающих *определению 3.3*. Этот этап требует $O(m^2n)$ сравнений на уровне пар слов (доказательство очевидно).

Действительно, для первой из возможных пар в худшем случае требуется $(m-1)n$ сравнений на предмет наличия синтагматической зависимости, для второй пары здесь будет $(m-2)n+(n-1)$ сравнение, для всех пар в рамках первой из СЭ-фраз, определяющих СЯУ, число сравнений здесь есть сумма $(n-1)$ первых членов убывающей арифметической прогрессии, равная $\frac{2n(m-1)-(n-2)}{2}(n-1)$, где первый член есть $(m-1)n$, разность равна -1. Для следующей СЭ-фразы задача выделения синтагматических зависимостей отличается числом оставшихся СЭ-фраз, в рамках которых ведётся поиск, меньшим на 2, поэтому общее число сравнений здесь будет уже $\frac{2n(m-3)-(n-2)}{2}(n-1)$. В итоге имеем убывающую арифметическую прогрессию с разностью $(n-1)\left(\frac{2n(m-3)-(n-2)-2n(m-1)+(n-2)}{2}\right)=-2n(n-1)$, общее же

число сравнений при выделении связей, отвечающих *определению 3.3*, для отдельной СЯУ есть сумма $\frac{m}{2}$ первых членов такой прогрессии и равняется $\frac{m}{4}(nm-n+2)$.

Итак, формирование *произвольных* отношений с применением принципа, изложенного в настоящем разделе, для отдельной СЯУ в общем случае предполагает порядка mn^2+m^2n операций типа сравнения. Отбор СЭ-фраз, модели линейных структур которых удовлетворяют *определению 3.4*, дополнительно требует порядка $2nm\left(\frac{m(n-1)}{2}\right)^2$ таких операций. Действительно, максимально возможное число связей, отвечающих *определению 3.3*, из расчёта того, что каждая связь появляется минимум в двух СЭ-фразах из определяющих СЯУ, оценивается как $\frac{m(n-1)}{2}$. В рамках отдельной фразы происходит сопоставление связей “каждая с каждой”, отдельное сопоставление формально есть сравнение наложением друг на друга двух подпоследовательностей индексов из модели линейной структуры этой фразы. Максимальная длина подпоследовательности равна n, сама подпоследовательность включает индексы, которые в модели линейной структуры лежат между двумя индексами, задающими связь. С учётом двух возможных направлений связи время поиска связей, отвечающих *определению 3.4*, для отдельной фразы оценивается как $2n\left(\frac{m(n-1)}{2}\right)^2$, для всех m СЭ-фраз в рамках СЯУ – как $2nm\left(\frac{m(n-1)}{2}\right)^2$.

В качестве примера рассмотрим выделение и классификацию синтаксических отношений на множестве вариантов правильного ответа для тестового задания открытой формы. Вопрос теста: *“Каковы негативные последствия переобучения при скользящем контроле?”*. В итоге было получено двадцать семь вариантов правильного ответа на данный вопрос (рис. 3.18). При этом основу формирования решетки $\Re fl$, представленной на рис. 3.19, составили максимально проективные ЕЯ-фразы (табл. 3.2) с минимумом слов, не нашедших прообразов по буквенному составу среди слов, составляющих фразы на рис. 3.18. Визуализа-

цию решетки диаграммой линий здесь и далее выполняет программная система “Concept Explorer” [28, 184], реализующая методы АФП. Содержательную интерпретацию решетка $\Re fl$ получает выделением морфологических классов слов на основе базиса импликаций, представленного на рис. 3.20.

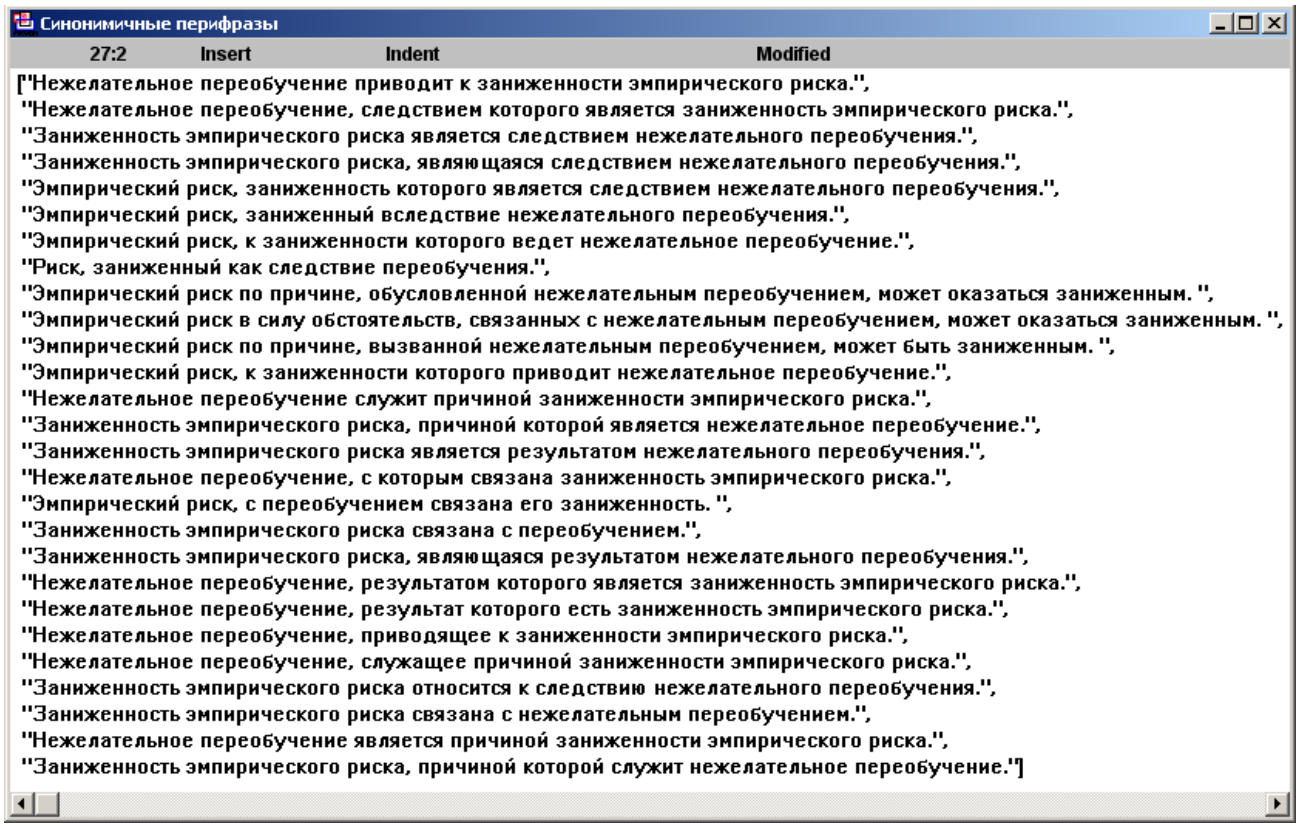


Рис. 3.18. Исходные данные для построения формального контекста (3.12)

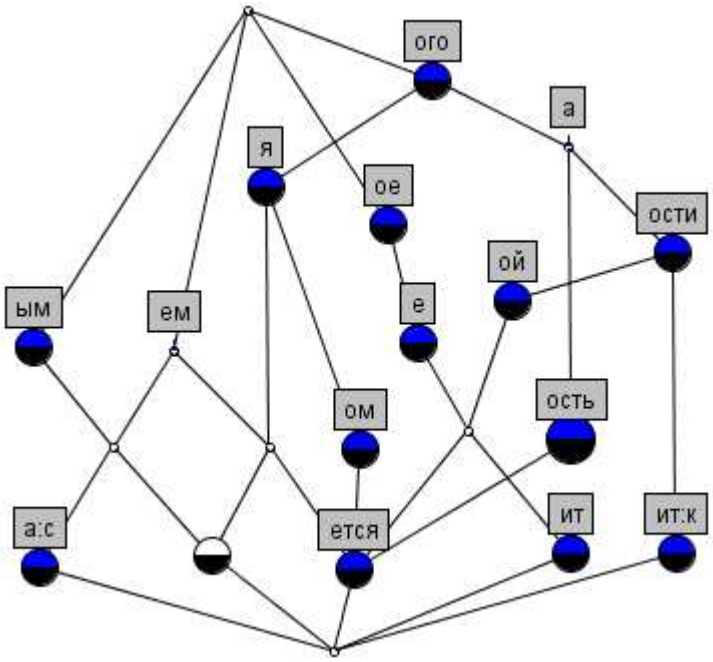


Рис. 3.19. Синтаксические отношения на основе сочетаний флексий

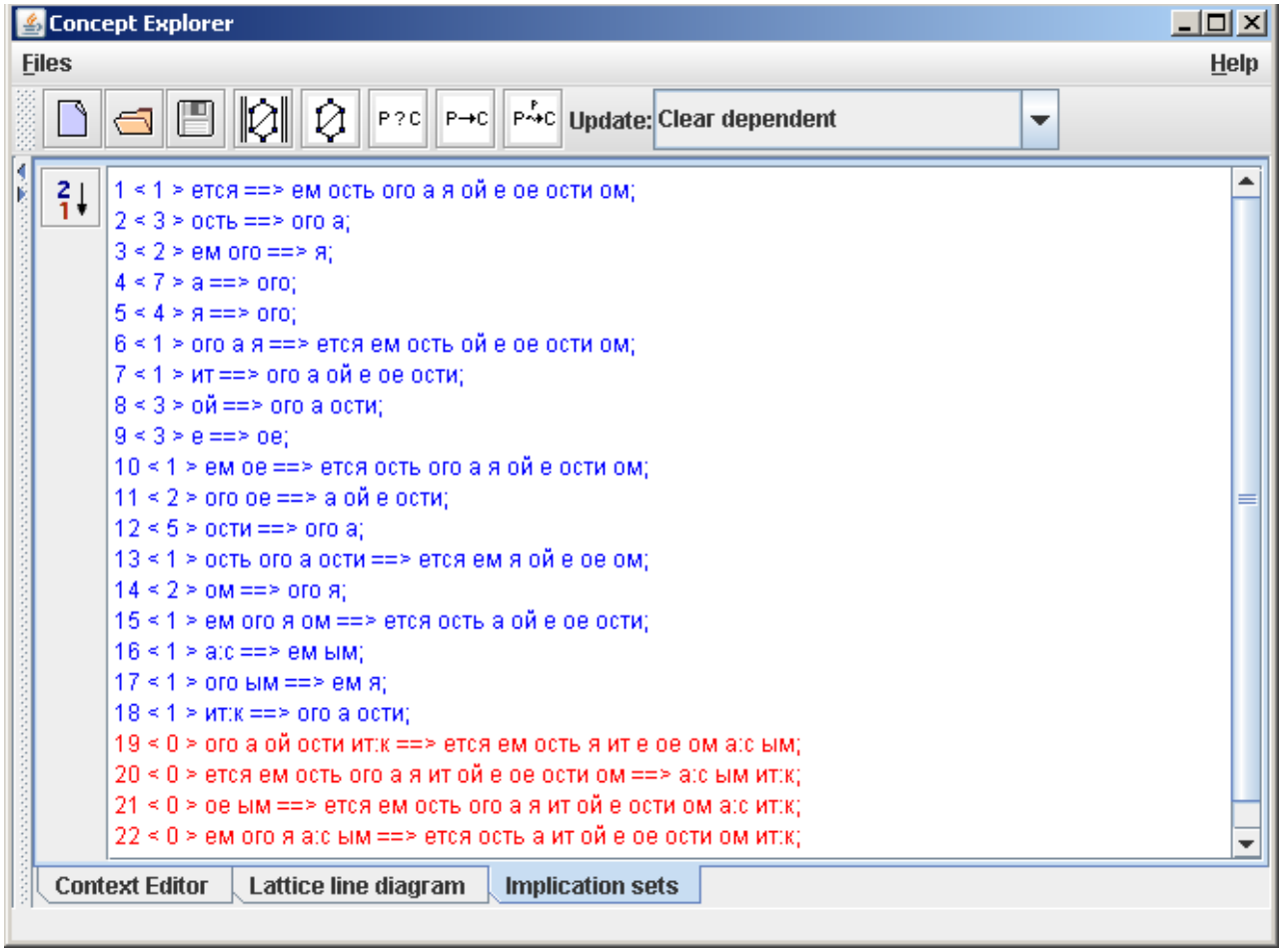


Рис. 3.20. Базис импликаций на основе результирующего множества ЕЯ-фраз

Таблица 3.2

Правильные ответы $Ts_i \in Ts'$

Основа	Флективная часть + предлог					
заниженн	ость	ости	ость	ости	ость	ости
эмпирическ	ого	ого	ого	ого	ого	ого
риск	а	а	а	а	а	а
нежелательн	ого	ое	ого	ое	ым	ое
переобучени	я	е	я	е	ем	е
явля	ется	–	ется	ется	–	–
следстви	ем	–	–	–	–	–
служ	–	ит	–	–	–	–
Причин	–	ой	–	ой	–	–
Результат	–	–	ом	–	–	–
Связан	–	–	–	–	а:с	–
Привод	–	–	–	–	–	ит:к

В примере на рис. 3.19 классы отношений соответствуют словоизменению прилагательных (*нежелательн-ого*, *эмпирическ-ого*) и существительных в составе генитивных конструкций (*результат-ом переобучени-я*, *следстви-ем пере-*

обучени-я). Последний в силу транзитивности синтаксического отношения в рамках последовательности соподчиненных слов может включать сочетания существительного (вне генитивных конструкций) с глаголом. Подробнее это отношение рассматривается в следующей главе работы.

Выводы

Таким образом, в третьей главе разработан принцип формирования и кластеризации семантических отношений выделением синтагматических зависимостей. Его программная реализация [113], представленная в **приложении 1** диссертации фрагментами исходного текста на языке Visual Prolog, позволяет выделять произвольные отношения в тексте в виде классов формальных понятий решётки. При этом синтаксические отношения рассматриваются как частный случай семантических и выражаются определенными сочетаниями флексий.

Наряду с описанием условий применимости для правил синонимических преобразований ГСС, введение характеристических функций для элементов толкований ЛЗ слов позволяет на основе формального контекста элементов толкования оценивать степень близости наборов таких правил, о которой говорилось в [33, 47, 52]. Тем не менее, описание смысла слова набором ХФ производится в шкале наименований. При обобщении утверждений независимых теорий одного и того же ЛЗ посредством отношения “или” не учитывается статистическая значимость каждого признака. Для введения в рассмотрение, к примеру, распределений значений ХФ здесь необходимо учитывать возможный контекст как толкуемого слова, так и тех слов, которые упоминаются в формализованном толковании (теории) его ЛЗ. Первостепенную роль здесь играет контекст существительного как база формирования отношений в рамках теории ЛЗ.

Семантике синтаксического контекста существительного, его использованию как основы кластеризации текстов, а также оценке информативности слов в составе такого контекста посвящается следующая глава работы.

Глава 4
СЕМАНТИЧЕСКАЯ КЛАСТЕРИЗАЦИЯ ТЕКСТОВ ЕСТЕСТВЕННОГО ЯЗЫКА ВЫДЕЛЕНИЕМ СИНТАКСИЧЕСКОГО КОНТЕКСТА СУЩЕСТВИТЕЛЬНОГО

Настоящая глава посвящена вопросам формирования знаний в рамках ситуаций языкового употребления по результатам разбора текстов внешней программой синтаксического анализа. На основе свойств соотношения смыслов соподчиненных слов решается задача выделения и классификации частичных смысловых эквивалентностей. Описывается использование синтаксического контекста имени существительного как основы выделения объектов и ситуаций, описываемых сравниваемыми текстами. Рассматривается критерий полезности решетки формальных понятий и его использование для определения силы семантической связи слов и в качестве основы систематизации конверсивов и расщепленных предикатных значений в рамках рассматриваемого синтаксического контекста. Основные результаты главы опубликованы в [87, 91, 93, 94, 167, 168].

4.1. Семантика синтаксиса как основа кластеризации

Как было показано в предыдущей главе, лексическая сочетаемость слова зависит от его семантического класса. Поэтому справедливо предположение о возможности выявления СК слова анализом его сочетаний с другими словами в ЕЯ-текстах по тематике заданной предметной области.

Следует отметить, что первостепенную роль для извлечения СК слова из набора текстов заданной тематики играет контекст целевого слова.

Наибольшую точность, как показывает практика, дают модели контекста на основе синтаксических связей в предложении [98, 120, 183].

В двух предыдущих главах основной акцент был уделен контексту предикатного слова, который определяется в первую очередь синтаксическими связя-

ми между предикатом и его семантическими актантами. Согласно постановке *задачи 1.1* для формализации понятий предметной области, обозначающих участников тех или иных ситуаций, необходимо ввести в рассмотрение сочетаемость соответствующих существительных со словами, являющимися синтаксически главными по отношению к ним. Причем наряду с сочетаниями “актант – предикат” требуется учитывать произвольные сочетания существительных в тексте между собой (в том числе посредством предлогов).

Как было показано в разделе 1.1, каждое выявляемое из текста понятие идентифицируется относительно некоторого множества отношений R, которые однозначно определяют СЯУ, формально представляемую посредством тройки (1.1). Поскольку сами отношения ассоциируются (в первую очередь) с предикатными словами – глаголами либо их производными, то наиболее приемлемым вариантом синтаксического контекста для существительного, называющего некоторое выявляемое понятие, будет последовательность соподчиненных слов (обозначение Sq – от англ. sequence):

$$Sq_{ki} = \{v_1, \ldots, v_{n(k,i)}, m_{ki}\}, \tag{4.1}$$

где v_1 – предикатное слово;

m_{ki} и $\forall v_l \in \{v_2, \ldots, v_{n(k,i)}\}$ есть некоторое существительное;

k – порядковый номер последовательности среди выявленных из текста Ts_i;

$n(k,i)$ – количество соподчиненных существительных последовательности.

Последовательности (4.1), выделяемые синтаксическим разбором СЭ-фраз исходного множества Ts, могут служить базой формирования множеств O и R в составе тройки (1.1). Основой выделения связей между понятиями $o_j \in O$ при этом являются синтаксические отношения R_q:

$$v_l R_q v_{l+1}, \ldots, v_{n(k,i)} R_q m_{ki}. \tag{4.2}$$

Тип отношения R_q характеризуется падежом зависимого слова и предлогом для связи главного и зависимого слова и, как было показано в разделе 3.5, соответствует имени синтагмы, которая определяет отношение R_q.

Наличие последовательности (4.1) в составе некоторого $Ts_i \in Ts$ делает возможным существование в любом другом тексте множества Ts последовательности $Sq_{lki} \neq Sq_{ki}$:

$$Sq_{lki} = \{v_l, m_{ki}\} \tag{4.3}$$

для $\forall v_l \in \{v_1, \ldots, v_{n(k,i)-1}\}$, где v_l связано с m_{ki} посредством отношения R_q. При этом обязательным является наличие $v_l R_q v_{l+1}$ в рамках последовательности (4.1). Будем называть последовательность Sq_{ki} ситуационным контекстом для m_{ki}. В этом случае Sq_{ki} в совокупности с множеством $\{Sq_{lki}\}_{l=1}^{n(k,i)-1}$ определяют некоторые отношения из R (либо ассоциируемые с ними понятия из O) относительно m_{ki}. Причем с любой Sq_{lki} связываются более абстрактные отношения, чем с Sq_{ki}.

Утверждение 4.1. При одновременном наличии последовательностей $Sq_{ki} = \{v_1, \ldots, v_{n(k,i)}, m_{ki}\}$ и $Sq_{1ki} = \{v_1, m_{ki}\}$ в разных текстах множества Ts имеет место частичная СЭ (относительно m_{ki}).

Пример: “Характеристика сложности семейства алгоритмов” ⇔ ”характеристика алгоритмов”. Подобная СЭ может задаваться, в частности, генитивной конструкцией [120, 183]. Для сравнения: “сложность подсемейства модели” ⇔ “сложность модели”.

Утверждение 4.2. При наличии отношения R_q между v_1 и v_2 возможно установление указанного отношения между v_1 и любым словом последовательности (4.1) вне зависимости от существующих отношений.

Доказательство следует из соотношения значений характеристических функций, задаваемых для соподчиненных слов в соответствии с *утверждением*

3.1. При этом для установления отношения R_q между v_1 и произвольным v_l, $l = 3,\ldots,n(k,i)$, а также между v_1 и m_{ki} зависимое слово должно быть приведено в соответствующую морфологическую форму.

Пример. Рассмотрим словосочетание *"рассматривать на множестве семейств алгоритмов"*. Допустимыми с точки зрения синтаксиса и семантики русского языка являются также словосочетания *"рассматривать на семействах"* и *"рассматривать на алгоритмах"*.

В настоящей работе в качестве базовой структуры для выявления и кластеризации понятий мы будем использовать ситуационные контексты вида (4.1), которые участвуют в описании частичных СЭ в соответствии с *утверждением 4.1*.

Ставится *задача*: путем синтаксического разбора предложений выявить указанные контексты в анализируемом тексте и на их основе выполнить концептуальную кластеризацию.

4.2. Концептуальная кластеризация текстов на основе результатов синтаксического разбора предложений

Результатом синтаксического анализа текста является набор деревьев разбора предложений. В настоящей работе синтаксический анализ осуществляется программой “Cognitive Dwarf” [110]. При тестировании данная программа показала самые точные результаты разбора.

На основе полученного набора деревьев формируются ситуационные контексты (4.1). При этом с каждого дерева последовательно считываются пары (x, y), где x – синтаксически главное слово, y – зависимое слово. Дальнейшая обработка считанных пар направлена на выявление последовательностей (4.1) и (4.3) в соответствии с *утверждением 4.1*. Обозначим множество последовательностей вида (4.1), формируемое относительно текста $Ts_i \in Ts$, как PS_i.

В качестве инструмента концептуальной кластеризации ситуационных контекстов (4.1) как основы выделения понятий будем использовать методы АФП, рассмотренные в предыдущих главах. Согласно постановке *задачи 1.1*, имеем формальный контекст:

$$K = (G,M,V,I), \tag{4.4}$$

где $G \supset Ts$; V – множество ситуаций, описываемых текстами из множества G; M – множество объектов и/или понятий, значимых в ситуациях из множества V; $I \subseteq G \times M \times V$.

Замечание. На основе *утверждения 4.2* справедливо будет утверждать, что $\forall v_l \in \{v_2,\ldots,v_{n(k,i)}\}$ в составе последовательности (4.1) обозначает некоторое понятие, значимое в ситуации v_1, наравне с m_{ki}. Таким образом если $V(Ts_i)$ рассматривать как множество ситуаций, описываемых текстом $Ts_i \in G$, а $M(Ts_i)$ – как соответствующее ему множество объектов согласно постановке *задачи 1.1*, то для любой Sq_{ki} $\{v_2,\ldots,v_{n(k,i)},m_{ki}\} \subset M(Ts_i)$. Причем $V(Ts_i) = \bigcup_k (Sq_{ki} \setminus \{m_{ki}\})$.

С учетом сказанного имеем расширение множеств $M(Ts_i)$ и $V(Ts_i)$ в соответствии с представленным ниже алгоритмом.

Алгоритм 4.1. Формирование троек-кандидатов на включение в отношение *I*.

Вход: PS_i; // множество последовательностей вида (4.1)

Выход: $PK_i = \{PK_{ki} : PK_{ki} = \{(g_i,m,v) : (g_i,m,v) \in I\}\}$;

// g_i есть некоторая символьная пометка для $Ts_i \in G$

Начало

$PK_i := \varnothing$; // Инициализация

Начало цикла. Пока $PS_i \neq \varnothing$

Выбрать Sq_{ki} из PS_i;

$PK_{ki} := \varnothing$;

Начало цикла. Для $l=1,\ldots,n(k,i)$

$PK_{ki}:=PK_{ki}\cup\{(g_i,m_{ki},v_l)\}$;

// $Sq_{ki}=\{v_1,\ldots,v_{n(k,i)},m_{ki}\}$ в соответствии с (4.1)

$j:=n(k,i)$;

Начало цикла. Пока $j>l$

$PK_{ki}:=PK_{ki}\cup\{(g_i,v_j,v_l)\}$;

$j:=j-1$;

Конец цикла {Пока $j>l$ *}*;

Конец цикла {Для $l=1,\ldots,n(k,i)$ *}*;

$PK_i:=PK_i\cup\{PK_{ki}\}$;

$PS_i:=PS_i\setminus\{S_{ki}\}$;

Конец цикла {Пока $PS_i\neq\varnothing$ *}*;

Конец {Алгоритм 4.1}.

При этом роль, в которой объект $m\in M(Ts_i)$ выступает относительно некоторой ситуации $v\in V(Ts_i)$, определяется типом отношения R_q между словом v и словом справа от него в последовательности (4.1). Поскольку указанный тип характеризуется падежом зависимого слова и предлогом для связи синтаксически главного и зависимого слова, то каждое $v\in V(Ts_i)$ в составе троек, формируемых *алгоритмом 4.1*, в зависимости от наличия/отсутствия предлога p_y между главным и зависимым словом представлено как

$$v=\begin{cases}x\bullet":"\bullet p_y\\ x\end{cases},$$

где x – синтаксически главное; y – зависимое слово; $\bullet$ – операция конкатенации. Для использования в дальнейших рассуждениях введем следующие функции:

$prep: v \to p_y$, которая ставит в соответствие каждому $v \in V(Ts_i)$ предлог для связи с зависимым словом; функцию $case: m \to c_y$, которая ставит в соответствие каждому именному $m \in M(Ts_i)$ символьное обозначение его падежа $c_y \in \{"nom","gen","dat","acc","ins","loc"\}$. Соответствие между словом и его начальной формой зададим с помощью функции $norm$.

Основные этапы построения решетки ФП $\Re(G, M, V, I)$ для формального контекста (4.4) представлены *алгоритмом 4.2*.

Алгоритм 4.2. Построение формального контекста для исходного множества текстов.

Вход: G ; // Исходное множество ЕЯ-текстов, $n(G) = |G|$

Выход: $K = (G, M, V, I)$; // Формальный контекст вида (4.4)

Начало

Шаг 1: Синтаксический анализ текстов из множества G с формированием множества PS_i для каждого $Ts_i \in G$;

Шаг 2: Для $\forall Ts_i \in G$ на основе PS_i выделить $M(Ts_i)$ и $V_1(Ts_i) \subset V(Ts_i)$: $V_1(Ts_i) = \{v_1 : \exists Sq_{ki} \in PS_i, Sq_{ki} = \{\{v_1, \ldots, v_{n(k,i)}, m_{ki}\}\}\}$;

Шаг 3: На основе выделенных $\{M(Ts_i) \mid i = \overline{1, n(G)}\}$ и $\{V_1(Ts_i) \mid i = \overline{1, n(G)}\}$ найти одноименные ситуации v, принадлежащие различным $V_1(Ts_i)$ и сходные по фигурирующим в них объектам $m \in M$: $M = \bigcup_i M(Ts_i)$ в сходных ролях;

Шаг 4: Приписать названиям ситуаций, выделенных на *Шаге 3*, одинаковые индексы в соответствующих $V_1(Ts_i)$ и PS_i;

Шаг 5: По аналогии с *Шагом 3* на основе PS_i найти разноименные ситуации v, принадлежащие различным $V_1(Ts_i)$ и сходные по фигурирующим в них объектам $m \in M$ в сходных ролях;

Шаг 6: По каждой выявленной на *Шаге 5* группе синонимов $Syn = \{v_1 : \exists Sq_{ki} = \{v_1, \ldots, v_{n(k,i)}, m_{ki}\}, i = \overline{1, n(G)}\}$ выделить канонический представитель v_1' с наибольшей частотой употребления и заменить все $v_1 \in Sq_{ki}$, $Sq_{ki} \in Syn$, на v_1';

Шаг 7: Выполнить *Шаги 3–6* для разноименных ситуаций, принадлежащих различным $V_1(Ts_i)$ и сходным по фигурирующим в них $m \in M$, но со сменой ролей (конверсивы);

Шаг 8: Для $\forall Ts_i \in G$ сформировать $V(Ts_i) = V_1(Ts_i) \cup \left(\bigcup_k (Sq_{ki} \setminus \{m_{ki}\} \setminus \{v_1\}) \right)$ и установить отношение I в соответствии с *алгоритмом 4.1* с учетом *Шагов 3–7*;

Конец {Алгоритм 4.2}.

Данный алгоритм описывает формирование множества ФП $\{(A, B) : A \subseteq G, B \subseteq M \times V, A = B', B = A'\}$ формального контекста (4.4). Здесь $V = \bigcup_i V(Ts_i)$, $M = \bigcup_i M(Ts_i)$ согласно введенным ранее обозначениям, A – объем, B – содержание формального понятия (A, B) согласно *определению 1.10*, причем $A' = \{(m, v) : m \in M, v \in V \mid \forall g \in A : m(g) = v\}$, $B' = \{g \in G \mid \forall (m, v) \in B : m(g) = v\}$. При этом решетка $\Re(G, M, V, I)$ дает требуемую классификацию текстов исходного множества G относительно описываемых текстами ситуаций и фигурирующих в этих ситуациях объектов.

4.3. Расщепленные предикатные значения и конверсивы в составе синтаксических контекстов существительных

При формировании множеств объектов и ситуаций на основе синтаксического анализа исходных текстов актуальна проблема наличия расщепленных значений в составе последовательностей (4.1).

В настоящей главе за основу механизма выявления РЗ мы возьмем правила синонимических преобразований типа замещения с расщеплением в рамках стандартных ЛФ. Фактически именно на эти правила мы ссылались в разделе 3.2 при формализации смыслового отношения в рамках расщепленного значения.

Пусть $\{Ts_1, Ts_2\} \subset G$ есть пара анализируемых текстов. Положим, что для Ts_1 выделено множество последовательностей вида (4.1) $SQ_1 = \{Sq_{k1} : Sq_{k1} \subset Ts_1\}$, $k = 1,\ldots,n(SQ_1)$, аналогично для Ts_2 $SQ_2 = \{Sq_{k2} : Sq_{k2} \subset Ts_2\}$, но при этом либо $k = 1,\ldots,n(SQ_1)$, либо $k = 1,\ldots,n(SQ_1)-1$, где $n(SQ_1) = |SQ_1|$

Утверждение 4.3. Применительно к паре (Ts_1, Ts_2) расщепленное предикатное значение $\{v_{11}, v_{12}\}$ будет иметь место в следующих двух случаях.

Случай 1.

$$\left.\begin{array}{l} Sq_{11} = \{v_{11}, v_{12}, v_{13}, \ldots, v_{1,idx(1,1)}, m_{11}\} \\ Sq_{21} = \{v_{11}, v_{12}, v_{23}, \ldots, v_{2,idx(2,1)}, m_{21}\} \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ Sq_{k1} = \{v_{11}, v_{12}, v_{k2}, \ldots, v_{k,idx(k,1)}, m_{k1}\} \\ \\ Sq_{k+1,1} = \{v_{11}, v_{k+1,2}, \ldots, v_{k+1,idx(k+1,1)}, m_{k+1,1}\} \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ Sq_{n(SQ_1),1} = \{v_{11}, v_{n(SQ_1),2}, \ldots, v_{n(SQ_1),idx(n(SQ_1),1)}, m_{n(SQ_1),1}\} \end{array}\right\},$$

$$\left.\begin{array}{l} Sq_{12} = \{v_{21}, v_{13}, \ldots, v_{1,idx(1,1)}, m_{11}\} \\ Sq_{22} = \{v_{21}, v_{23}, \ldots, v_{2,idx(2,1)}, m_{21}\} \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ Sq_{k2} = \{v_{21}, v_{k2}, \ldots, v_{k,idx(k,1)}, m_{k1}\} \\ Sq_{k+1,2} = \{v_{21}, v_{k+1,2}, \ldots, v_{k+1,idx(k+1,1)}, m_{k+1,1}\} \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ Sq_{n(SQ_1),2} = \{v_{21}, v_{n(SQ_1),2}, \ldots, v_{n(SQ_1),idx(n(SQ_1),1)}, m_{n(SQ_1),1}\} \end{array}\right\}.$$

Случай 2.

$$\left.\begin{array}{l} Sq_{11}=\{v_{11},v_{13},\ldots,v_{1,idx(1,1)},m_{11}\} \\ Sq_{21}=\{v_{11},v_{23},\ldots,v_{2,idx(2,1)},m_{21}\} \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ Sq_{k-1,1}=\{v_{11},v_{k-1,2},\ldots,v_{k-1,idx(k-1,1)},m_{k-1,1}\} \\ \\ Sq_{k1}=\{v_{11},v_{12}\} \\ \\ Sq_{k+1,1}=\{v_{11},v_{k+1,2},\ldots,v_{k+1,idx(k+1,1)},m_{k+1,1}\} \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ Sq_{n(SQ_1),1}=\{v_{11},v_{n(SQ_1),2},\ldots,v_{n(SQ_1),idx(n(SQ_1),1)},m_{n(SQ_1),1}\} \end{array}\right\},$$

$$\left.\begin{array}{l} Sq_{12}=\{v_{21},v_{13},\ldots,v_{1,idx(1,1)},m_{11}\} \\ Sq_{22}=\{v_{21},v_{23},\ldots,v_{2,idx(2,1)},m_{21}\} \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ Sq_{k-1,2}=\{v_{21},v_{k-1,2},\ldots,v_{k-1,idx(k-1,1)},m_{k-1,1}\} \\ Sq_{k+1,2}=\{v_{21},v_{k+1,2},\ldots,v_{k+1,idx(k+1,1)},m_{k+1,1}\} \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ Sq_{n(SQ_2),2}=\{v_{21},v_{n(SQ_1),2},\ldots,v_{n(SQ_1),idx(n(SQ_1),1)},m_{n(SQ_1),1}\} \end{array}\right\}.$$

Здесь функция $idx(k,i)$ возвращает максимальное значение второго индекса при v в заданной последовательности Sq_{ki}, а $n(SQ_2)=n(SQ_1)-1$.

Замечание. С учетом возможного наличия конверсивов слова v_{21}, применительно к обоим указанным случаям РПЗ предполагается, что соответствующая замена уже выполнена, а SQ_1 и SQ_2 описывают одно и то же множество объектов относительно одной и той же ситуации, обозначаемой посредством v_{21}, то есть без мены ролей.

Для использования в дальнейших рассуждениях введем функцию $Spv:(v_{11},v_{12})\to v_{21}$, которая ставит в соответствие расщепленному предикатному значению $\{v_{11},v_{12}\}$ его однословное выражение v_{21}.

Множество РПЗ, определяемых *утверждением 4.3*, включает в себя расщепления с глаголом-связкой, а также расщепления с глаголами – синтаксическими оформителями ситуаций, обозначаемых именами существительными, представляющими собой языковое обозначение ролей участников ситуаций.

Обобщая введенное формальное определение РПЗ, дадим теперь понятие конверсива, опираясь на описанные И.А. Мельчуком правила синонимических преобразований типа конверсивных замещений [62, с. 152–153].

Пусть SQ_1 и SQ_2 – пара множеств последовательностей вида (4.1).

Утверждение 4.4. Применительно к $\{SQ_1, SQ_2\}$ имеет место конверсив, если для $\forall\, Sq_{k1} \in SQ_1$ найдется последовательность $Sq_{j2} \in SQ_2$ такая, что при этом могут иметь место следующие случаи взаимного соответствия Sq_{k1} и Sq_{j2}.

Случай 1.

$$Sq_{k1} = \{v_{11}', v_{k2}, v_{k3}, \ldots, v_{k,idx(k,1)}, m_{k1}\},$$

$$Sq_{j2} = \{v_{21}', v_{k2}', v_{k3}, \ldots, v_{k,idx(k,1)}, m_{k1}\}.$$

При этом $norm(v_{11}') = norm(v_{21}')$, $norm(v_{k2}) = norm(v_{k2}')$, причем в общем случае $prep(v_{11}') \neq prep(v_{21}')$, а $case(v_{k2}) \neq case(v_{k2}')$.

Случай 2.

$$Sq_{k1} = \{v_{11}', v_{12}', v_{k2}, v_{k3}, \ldots, v_{k,idx(k,1)}, m_{k1}\},$$

$$Sq_{j2} = \{v_{21}', v_{k2}', v_{k3}, \ldots, v_{k,idx(k,1)}, m_{k1}\}.$$

Здесь $norm(v_{k2}) = norm(v_{k2}')$, $case(v_{k2}) \neq case(v_{k2}')$ (в общем случае), но при этом для Sq_{j2} $\exists\, Sq_{k1}' \in SQ_1$: $\{Sq_{k1}', Sq_{j2}\}$ соответствует *случаю 1*, $Sq_{k1}' \neq Sq_{k1}$, а для Sq_{k1} $\exists\, Sq_{j2}' \in SQ_2$: $\{Sq_{k1}, Sq_{j2}'\}$ также удовлетворяет требованию *случая 1* настоящего *утверждения и* $Sq_{j2}' \neq Sq_{j2}$.

Замечание. Положим $v_{21} = norm(v_{21}')$ в Sq_{j2} для *случая 1* и *случая 2*, $v_{11} = norm(v_{11}')$ и $v_{12} = norm(v_{12}')$ в Sq_{k1} для *случая 2* соответственно. По аналогии с РПЗ будем называть пару $\{v_{11}, v_{12}\}$ расщепленным конверсивом для v_{21}.

Определяемые *утверждением 4.4* конверсивные замены включают в себя как простые перестановки актантов исходного слова на другие места без расщепления последнего, так и замены РПЗ на их нерасщепленные семантические эквиваленты с последующей перестановкой актантов. В частности, в качестве замен без расщепления могут быть рассмотрены синонимические замещения. Здесь для *случая 1* мы имеем: $k = j$, $prep(v_{11}') = prep(v_{21}')$, а $case(v_{k2}) = case(v_{k2}')$. *Актуальными* здесь *являются* автоматическая лингвистически интерпретируемая классификация выявляемых конверсивов и определение порядка их замен в анализируемых текстах.

Для установления порядка применения конверсивных преобразований воспользуемся следующими эвристическими правилами.

Правило 1. При выборе возможного варианта конверсивной замены без расщепления предпочтение отдается слову с минимальной многозначностью. При этом степень многозначности количественно определяется числом найденных для рассматриваемого слова предикатных лексических значений.

Правило 2. При нескольких вариантах замен на слова с одинаковым числом возможных предикатных лексических значений предпочтение отдается слову с максимальным числом беспредложных валентностей.

Замечание. Как отметил академик Ю.Д. Апресян [3, с. 149], беспредложные падежи выступают в качестве обязательных чаще, чем предложные, прямой – чаще, чем косвенные. Данный факт дает основание предположить о том, что из конверсивного ряда более компактное описание ситуации (более четкое выражение смысла) характерно для того предикатного слова, у которого число беспредложных валентностей максимально.

Правило 3. При наличии нескольких вариантов замены расщепленного конверсива нерасщепленным семантическим эквивалентом следует руководствоваться *правилом 1* и *правилом 2* для конверсивных замен без расщепления.

Правило 4. Если для найденного по *правилу 3* семантического эквивалента расщепленного конверсива существуе вариант замены по *правилу 1* либо *правилу 2*, то следует производить замену расщепленного конверсива именно на этот вариант.

Для решения задачи лингвистически интерпретируемой классификации конверсивов, выявляемых в соответствии с *утверждением 4.4* на основе вышеуказанных *правил 1–4*, будем использовать уже рассмотренные методы АФП.

Введем в рассмотрение формальный контекст:

$$Kconv = (Gconv, Mconv, Iconv), \tag{4.5}$$

в котором согласно *утверждению 4.4*

$$Gconv = \left\{ v_{21} : v_{21} = norm\left(v_{21}{}'\right) \right\},$$

$$Mconv = \left\{ vconv : vconv = \begin{cases} v_{11} \\ v_{12} \bullet ":" \bullet v_{11} \end{cases} \right\},$$

где $v_{11} = norm\left(v_{11}{}'\right)$;

$v_{12} = norm\left(v_{12}{}'\right)$; операция конкатенации имеет место для *случая 2* из рассматриваемых *утверждением 4.4*; отношение $Iconv \subseteq Gconv \times Mconv$ ставит в соответствие каждому варианту конверсивной замены $v_{21} \in Gconv$ заменяемый конверсив $vconv \in Mconv$.

Пусть $\Re conv$ есть решетка ФП для контекста (4.5). Введем индексы: 1 – для контекстов вида (3.12) и (4.5), формируемых с применением методики выделения и классификации синтаксических отношений, предложенной в разделе 3.5; 2 – для контекстов тех же видов, но формируемых на основе синтаксического разбора ЕЯ-фраз программой “Cognitive Dwarf”. Положим, что решетки $\Re conv_2$ и $\Re f_2$ формируются на основе неструктурированного текста заданной тематики,

включающего подмножество множества Ts относительно языкового контекста ситуации (1.1). Мощность этого подмножества зависит от репрезентативности текста. Под показателем репрезентативности здесь следует понимать количество форм языкового описания заданной ситуации, присутствующих в анализируемом тексте и использованных при формировании $\Re f_1$ и $\Re conv_1$.

Каждая область решетки для формального контекста (4.5) вне зависимости от исходных данных для построения при единственности НОПП и НОСП получает содержательную интерпретацию группы смысловых отношений со сходным составом аргументов и сходным характером перестановок аргументов (типом конверсии).

Введем в рассмотрение базисы импликаций: $Lconv_1$ – базис импликаций для формального контекста $Kconv_1$, $Lconv_2$ – для формального контекста $Kconv_2$ соответственно.

Утверждение 4.5. Будем считать классификацию отношений из R в (1.1) на основе контекста (3.12) допустимой применительно к случаю наличия в Ts фраз, отвечающих условиям *утверждения 4.4*, если $\Re f_1 \subset \Re f_2$ и $\exists\left(PRconv_1 \to CSconv_1\right) \in Lconv_1$, при этом $\exists\left(PRconv_2 \to CSconv_2\right) \in Lconv_2$, где $PRconv_1 \cap PRconv_2 \neq \varnothing$, а $CSconv_1 \cap CSconv_2 \neq \varnothing$.

При этом случай $\Re f_1 = \Re f_2$ не обязательно соответствует тексту с максимальной репрезентативностью по сформулированному нами критерию. Встречаемость тех или иных сочетаний флексий находится в зависимости и от количества описываемых текстом ситуаций. В частности, текстом может описываться несколько ситуаций, близких рассматриваемой по составу участников и их ролевой ориентации.

Вопросам взаимосвязи качественных характеристик решеток ФП и информативности отдельного признака в текстовой классификации посвящается следующий раздел.

4.4. Информативность признака и критерий полезности решётки формальных понятий

Используемое для формирования моделей (4.4) и (4.5) множество текстов представляет собой тематическое подмножество того текстового корпуса, который по жанровому разнообразию представленного в нем рода словесности [112] следует отнести к научной прозе. Рассмотрим, каким образом особенности исходных текстов влияют на качество концептуальной кластеризации, выполняемой методами АФП.

Вначале сформулируем более общее определение понятия репрезентативности, введенного в предыдущем разделе.

Определение 4.1. Под репрезентативностью множества текстов будем понимать способность этого множества отображать все свойства предметной области, релевантные для некоторого заданного лингвистического исследования.

При использовании последовательностей вида (4.1) в качестве основы кластеризации выбираемая оценка репрезентативности для исходного текстовго материала должна стать основой практических выводов как оносительно точности алгоритмов синтаксического анализа, так и направлениях их дальнейшего совершенствования. В этом плане естественной оценкой репрезентативности может послужить суммарная частота Fs, с которой последовательности вида (4.1), соответствующие условию *утверждения 4.1*, встречаются в анализируемых текстах. Но с учетом отсутствия ограничений на тип отношения R_q между словами в (4.1) за указанную оценку следует принять отношение частоты Fs к количеству n_q типов отношений R_q в рамках последовательностей вида (4.1):

$$Fq = \frac{Fs}{n_q} = \frac{n_S}{nn_q}, \tag{4.6}$$

где n_S – количество последовательностей вида (4.1), извлеченных из анализируемого множества текстов; n – число слов в анализируемом множестве текстов.

Хорошим примером репрезентативности текста в соответствии с критерием (4.6) с характерной минимизацией n_q при максимизации Fs может послужить обзорная статья [9]. На рис. 4.1 представлена решетка ФП для указанного текста. Соответствующий ей формальный контекст $Kv = (Gv, Mv, Iv)$ можно представить как получаемый из формального контекста вида (4.4), в котором множество G представлено единственным элементом – некоторой символьной пометкой gv для рассматриваемого текста. При этом

$$Gv = \{m \in M : \exists v \in V, (gv, m, v) \in I\},$$

$$Mv = \{v \in V : \exists m \in M, (gv, m, v) \in I\},$$

$$Iv = \{(m, v) : (gv, m, v) \in I\}.$$

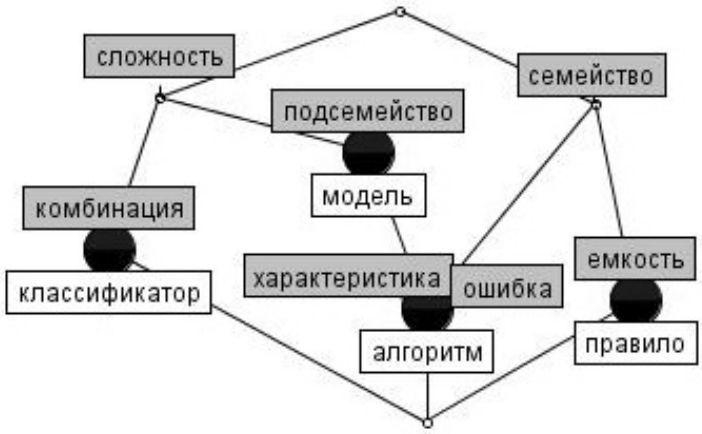


Рис. 4.1. Пример решетки ФП для множества ситуационных контекстов

Репрезентативность текстового материала в значительной мере влияет на способность решетки ФП выделять общие свойства классифицируемых объектов и соответствие формируемой решетки требованию иерархичности лексических ресурсов.

С целью достижения указанных требований для решетки в работе [183] был предложен критерий полезности. Если A_i – объем, B_i – содержание фор-

мального понятия (A_i, B_i) согласно *определению 1.10*, то данный критерий следует рассматривать как коэффициент F :

$$F = \max_{j=1}^{J} \left(\sum_{i=1}^{n_j} |A_i| \right), \tag{4.7}$$

где J – индексное множество цепочек; $j \in J$ – номер цепочки; n_j – количество ФП в цепочке с номером *j*; i – порядковый номер ФП в цепочке.

Максимизация указанного критерия при генерации формального контекста вида (4.5), в частности, предполагает выбор пар $\{v_{21}, vconv\}$ таким образом, чтобы любое ФП $Cconv = (Aconv, Bconv)$ в решетке $\Re conv(Gconv, Mconv, Iconv)$ входило в цепочку максимальной длины при $|Aconv| \to \max$.

При этом само формирование решетки ведется по областям. Вначале на основе групп подряд идущих последовательностей вида (4.1) на выходе синтаксического анализа *алгоритмом 4.3* выявляются пары соподчиненных слов, задающих РПЗ и расщепленные конверсивы в соответствии с условиями *утверждений 4.3* и *4.4*. Этим же алгоритмом производится замена найденных РПЗ и конверсивов на их однословные выражения согласно *правилам 1–4* во всех исходных последовательностях соподчиненных слов для последующего использования указанных последовательностей в качестве исходных данных *алгоритма 4.1*. Функция $Conv: vconv \to v_{21}$, упоминаемая в *алгоритме 4.3*, есть обобщение функции $Spv: (v_{11}, v_{12}) \to v_{21}$, введенной нами ранее для расщепленных предикатных значений, выявляемых в соответствии с *утверждением 4.3*. При этом

$$vconv = \begin{cases} v_{11} \\ v_{12} \bullet ":" \bullet v_{11} \end{cases} \tag{4.8}$$

согласно разделению множества признаков формального контекста вида (4.5).

Алгоритм 4.3. Формирование кандидатов на включение в отношение $Iconv$.

Вход: $PS = \left\{PS_i : PS_i = \left\{Sq_{ki} : Sq_{ki} = \left\{v_1, \ldots, v_{n(k,i)}, m_{ki}\right\} \middle| \; i = \overline{1, n(G)}\right\}\right\}$;

Выход: PC; // Множество объектов с наборами признаков

$Pconv = \{(vconv, v_{21}) : v_{21} = Conv(vconv)\}$;

PSC; // Множество, полученное заменой РПЗ и конверсивов во всех $Sq_{ki} \in PS_i$ из исходного PS

Начало

$PC := \varnothing$; $Pconv := \varnothing$; // Инициализация

Начало цикла. Для $i = 1, \ldots, n(G)$

Сформировать множество PS'_i из групп $PS_{ki} \subseteq P_i$ подряд идущих Sq_{ki} с одним и тем же v_1;

Конец цикла {Для $i = 1, \ldots, n(G)$*}*;

$PS' := \left\{PS'_i \middle| \; i = \overline{1, n(G)}\right\}$;

Начало цикла. Для всех PS'_i таких, что $i = \overline{1, n(G)}$

Выбрать $PS'_j \in PS' : \; j \neq i$;

Начало цикла. Для всех $PS_{k1i} \in PS'_i$

Найти $PS_{k2i} \in PS'_j : \left\{PS_{k1i}, PS_{k2j}\right\}$ удовлетворяет условию *Утверждения 4.4*;

$Pconv := Pconv \cup \{(vconv, v_{21})\}$ согласно (4.8);

Если $PC = \varnothing$ *то*

$PC_k := \{vconv\}$;

$PC := PC \cup \{(v_{21}, PC_k)\}$;

иначе

Найти $(v_{21}, PC_k) \in PC$;

$PC := PC \setminus \{(v_{21}, PC_k)\}$;

$PC_k := PC_k \cup \{vconv\}$;

$PC := PC \cup \{(v_{21}, PC_k)\}$;

Конец {Если $PC = \varnothing$ *}*;

Конец цикла {Для всех $PS_{k1i} \in PS_i'$ *}*;

Конец цикла {Для всех PS_i' *таких, что* $i = \overline{1, n(G)}$ *}*;

$PSC := \varnothing$;

Начало цикла. Для всех PS_i таких, что $i = \overline{1, n(G)}$

$PSC_i := \varnothing$;

Начало цикла. Для всех $Sq_{ki} \in PS_i$

Если $\exists(vconv, v_{21}) \in Pconv$ *и* $vconv \in Sq_{ki}$

то сформировать $Sqsc_{ki}$ заменой *vconv* на v_{21} в Sq_{ki} согласно *правилам 1–4*;

иначе $Sqsc_{ki} := Sq_{ki}$

Конец {Если $\exists(vconv, v_{21}) \in Pconv$ *и* $vconv \in Sq_{ki}$ *}*;

$PSC_i := PSC_i \cup \{Sqsc_{ki}\}$;

Конец цикла {Для всех $Sq_{ki} \in PS_i$ *}*;

$PSC := PSC \cup \{PSC_i\}$;

Конец цикла {Для всех PS_i *таких, что* $i = \overline{1, n(G)}$ *}*;

Конец {Алгоритм 4.3}.

Отдельная цепочка $PCch_j$ формируется на основе множества PC объектов с заданными наборами признаков согласно *алгоритму 4.4*. С целью минимизации числа спорных ФП каждое следующее ФП в цепочке выбирается по принципу постепенного уменьшения содержания и максимизации числа общих признаков с потенциальным подпонятием при минимуме общих признаков с любым ФП, не входящим в цепочку.

Алгоритм 4.4. Формирование цепочки в $\Re conv$ по максимуму критерия (4.7).

Вход: PC на выходе *алгоритма 4.3*;

Выход: $PCch_j = \{(v_{21}, PC_k) \mid (v_{21}, PC_k) \in PC, \leq\}$; // PC_k – набор признаков для v_{21}

PR; // Подмножество исходного PC, не вошедшее в $PCch_j$

$PCneigh_j$; // Множество ФП, соседних для связанных отношением $\leq$

Начало

$PCch_j := \varnothing$;

$PCneigh_j := \varnothing$; // Инициализация

Выбрать $(v_{\max}, PC_{\max})$ из PC: $|PC_{\max}| \to \max$;

$PC := PC \setminus \{(v_{\max}, PC_{\max})\}$;

$PCch_j := PCch_j \cup \{(v_{\max}, PC_{\max})\}$;

$PC_{tmp} := PC_{\max}$;

Начало цикла

Выбрать (v_{21}, PC_k) из $PC: PC_k \subset PC_{tmp}$ и $|PC_{tmp} \cap PC_k =: Cr| \to \max$;

При $Cr = \varnothing$ *выход из цикла*;

$PC_{tmp} := PC_k$;

$PCch_j := PCch_j \cup \{(v_{21}, PC_k)\}$;

$PC := PC \setminus \{(v_{21}, PC_k)\}$;

Выбрать $\{(v_{Cr}, PC_{Cr}) \mid PC_{Cr} \supseteq Cr\} =: PCr \subseteq PC$;

$PCneigh_j := PCneigh_j \cup PCr$;

$PC := PC \setminus PCr$;

Конец цикла;

$PR := PC$;

Конец {Алгоритм 4.4}.

Алгоритмом 4.5 строится множество цепочек для формальных понятий из множества $PCneigh_j$. Множество $PCneigh_j$ есть в соответствии с *определением 1.17* множество формальных понятий, соседних по отношению к тем ФП $Cconv = (Aconv, Bconv)$, между которыми устанавливается отношение ≤ при формировании цепочки. Здесь $Aconv = \{v_{21}\}$, $Bconv = PC_k$.

Алгоритм 4.5. Генерация множества цепочек для "соседних" ФП в решетке $\Re conv$.

Вход: PC на выходе *алгоритма 4.3*;

Выход: $PCch = \{PCch_j : PCch_j = \{(v_{21}, PC_k) : (v_{21}, PC_k) \in PC, \leq\}\}$;

Начало

$PCch := \varnothing$; // Инициализация

Начало цикла

Сформировать $PCch_j$, $PCneigh_j$ и PR

алгоритмом 4.4 на основе PC;

При $|PCch_j| \leq 1$ *выход из цикла*;

$PCch := PCch \cup \{PCch_j\}$;

$PC := PCneigh_j \cup PR$;

Конец цикла;

Конец {Алгоритм 4.5}.

Немаловажную роль при максимизации критерия (4.7) для решетки ФП играет инофрмативность каждого признака. Как было показано в [183], информативность признака тем ниже, чем большим числомом объектов рассматриваемого формального контекста он разделяется.

При построении $\Re conv$ с применением *алгоритмов 4.3–4.5* значимость неинформативных признаков будет минимальной согласно *правилу 1* порядка при-

менения конверсивных преобразований (доказательство очевидно). Поэтому справедливым будет утверждать, что множество формальных понятий формального контекста $Kconv = (Gconv, Mconv, Iconv)$ есть $\bigcup_{j=1}^{J} PCch_j$ на выходе *алгоритма 4.5*.

На рис. 4.2 представлен пример решетки $\Re conv$, построенной с применением *алгоритмов 4.3–4.5*. В качестве экспериментального текстового материала были взяты варианты ответов на тестовые задания открытой формы по материалам статьи [9]. Область в решетке, отвечающая условию *утверждения 4.5*, обозначена прямоугольником. Для сравнения на рис. 4.3 показана аналогичная решетка, полученная для примера из табл. 3.2 в соответствии с *теоремой 3.3*.

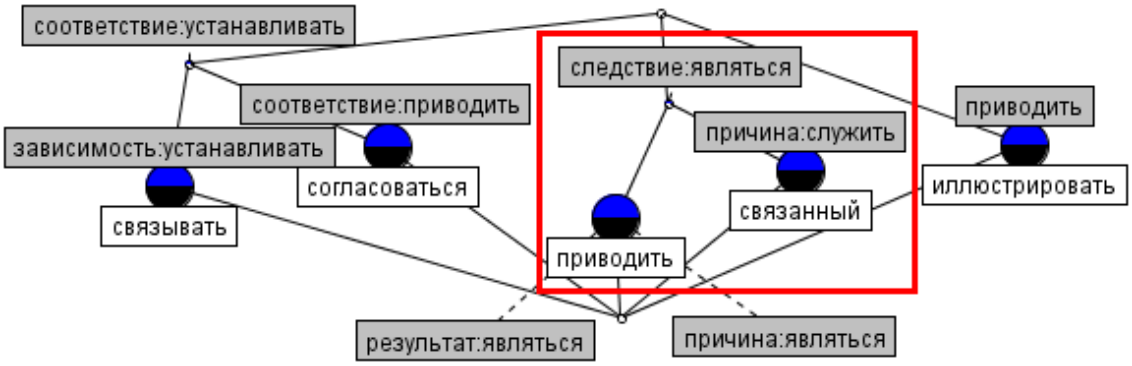


Рис. 4.2. Группировка РПЗ и конверсивных замен по результатам Cognitive Dwarf

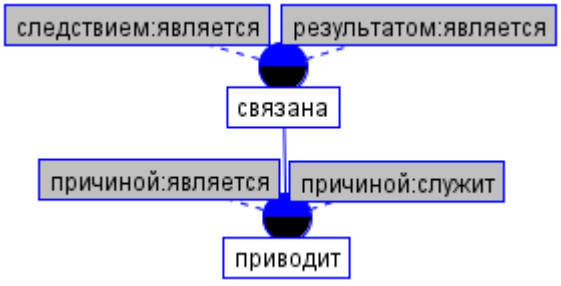


Рис. 4.3. РПЗ и конверсивы в составе фраз из T' (табл. 3.1)

Рассмотрим теперь решетку $\Re v(Gv, Mv, Iv)$ для множества ситуационных контекстов вида (4.1), пример которой представлен на рис. 4.1, в плане максимизации критерия (4.7). При отборе признаков, которыми будут характеризоваться объекты в составе множества Gv, в целях минимизации влияния неинформативных

признаков на вычисляемое значение критерия (4.7) для решетки $\Re v$ следует учитывать частоту $Cnt(v)$, с которой в анализируемом тексте потенциальный признак v встречается с различными $m \in Gv$.

Пусть $PCnt$ есть множество пар вида $(v, Cnt(v))$ для каждого признака множества Mv. Положим, что множество PCV есть аналог множества PC на выходе *алгоритма 4.3* и содержит пары вида "объект – набор признаков" для формального контекста $Kv = (Gv, Mv, Iv)$. Введем также в рассмотрение $PCVch$ – аналог множества $PCch$, формируемого *алгоритмом 4.5*. Тогда формирование контекста Kv с исключением из рассмотрения малоинформативных признаков можно представить с помощью следующего алгоритма.

Алгоритм 4.6. Генерация формального контекста Kv.

Вход: $PS_i = \{Sq_{ki} : Sq_{ki} = \{v_1, \ldots, v_{n(k,i)}, m_{ki}\}\}$;

Выход: $Kv = (Gv, Mv, Iv)$;

Начало

Сформировать PCV на основе PS_i;

Сформировать $PCnt$;

$\Delta_F := 0$;

Начало цикла. Пока $\Delta_F \le 0$

$\Delta_F := |\Delta_F|$;

Сформировать $PCVch$ на основе PCV;

$F_{tmp} := \max_{j=1}^{JV}(|PCVch_j : PCVch_j \in PCVch|)$;

// JV – индексное множество цепочек относительно решетки $\Re v$

$\Delta_F := \Delta_F - F_{tmp}$;

Найти $v_C \in Mv$: $(v_C, Cnt(v_C)) \in PCnt$ и $Cnt(v_C)$ – максимально;

Начало цикла. Для всех $(m, PCV_k) \in PCV$

$PCV_k := PCV_k \setminus \{v_C\};$

Конец цикла {Для всех $(m, PCV_k) \in PCV$ *}*;

$PCnt := PCnt \setminus \{(v_C, Cnt(v_C))\};$

Конец цикла {Пока $\Delta_F \le 0$ *}*;

$Kv := \bigcup_{j=1}^{JV} PCVch_j$;

Конец {Алгоритм 4.6}.

Следует отметить, что зависимость вероятности, с которой подпоследовательность слов из структуры (4.1), выделяемая согласно *алгоритму 4.1* при формировании пар “объект – признак”, будет подчиняться некоторому другому слову этого же синтаксического контекста в рассматриваемом корпусе текстов, от вероятностей появления в корпусе этого слова и подпоследовательности отдельно друг от друга *алгоритмом 4.6* не учитывается. Причина заключается во взаимной зависимости составов таких подпоследовательностей, вытекающей из *утверждения 4.2*, при их употреблении в тексте за рамками синтаксического контекста (4.1). Использование мер информативности различных комбинаций слов из (4.1) с учетом указанной зависимости, а также отсутствия ограничений на тип синтаксического отношения между соподчиненными словами – тема отдельного исследования.

Выводы

Таким образом, в четвёртой главе принцип формирования и экспериментальной оценки знаний в виде классов СЭ согласно постановке *задачи 1.1* развит применительно к наличию конверсивов и РПЗ в анализируемых текстах. Критерием выбора возможного варианта замены конверсива либо РПЗ здесь является минимум многозначности при максимальном числе беспредложных смысловых валентностей слова, на которое производится замена. При этом степень многозначности определяется числом СЯУ, в которых фигурирует слово.

Наряду с выделением семантических отношений рассмотрение синтаксического контекста существительного в качестве базовой структуры семантической кластеризации позволяет решить задачу автоматического извлечения элементов толкования лексического значения непосредственно из текстов. Сказанное дает возможность формирования прецедентов для ситуаций ЛФ-синонимии также на основе множеств текстов, в каждом из которых все тексты семантически эквивалентны друг другу.

Применительно к множеству выявляемых синтаксических контекстов существительных рассмотренный в заключительном разделе главы критерий полезности решетки формальных понятий позволяет делать выводы о силе семантической связи слов в рамках указанных контекстов. К примеру, чем в большем числе синтаксических контекстов фигурирует заданное предикатное слово, тем менее однозначно оно определяет существительное, ему подчиненное, и, следовательно, тем меньше сила их семантической связи [183], что означает и меньшее значение полезности решетки для множества ситуационных контекстов в соответствии с *алгоритмом 4.6*. Сказанное актуально при выборе возможного варианта замены некоторого конверсива либо РПЗ в задаче минимизации оптимального слова в языке сети Петри, построенной из примитивов вида (2.8). Кроме того, значение критерия полезности решетки ФП для совокупности РПЗ дает возможность делать выводы о сходстве ролевого состава ситуаций, обозначаемых в составе расщепленных предикатных значений словами-аргументами той или иной лексической функции.

В следующей главе будет рассмотрено, каким образом на основе синтаксического контекста существительного определяется количественная оценка схожести ситуаций языкового употребления, порождаемых независимо друг от друга, а также перспективы использования свойств указанного контекста в задаче сжатия информации при построении текстовых баз данных по заданной предметной области.

Глава 5
МЕТОД ЧИСЛЕННОЙ ОЦЕНКИ СЕМАНТИЧЕСКОЙ СХОЖЕСТИ ТЕКСТОВ ПРЕДМЕТНОГО ЯЗЫКА

В данной главе рассматриваются вопросы использования меры схожести формальных понятий в решетке применительно к ситуациям языкового употребления. Описывается методика построения формального контекста СЯУ по результатам синтаксического разбора совокупности семантически эквивалентных фраз предметно-ограниченного естественного языка (с учётом возможного наличия расщеплённых предикатных значений). Для тезауруса предметной области вводится представление формальным контекстом и ориентированное на него объектно-признаковое описание отдельной СЯУ как тезаурусной единицы. Вводится количественная оценка схожести формальных контекстов ситуаций языкового употребления. Описываются правила установления семантической эквивалентности фраз предметно-ограниченного ЕЯ-подмножества. Основные результаты главы опубликованы в [63, 86, 87, 172, 173].

5.1. Синтаксические и семантические связи в ситуации языкового употребления

В разделе 3.5 было рассмотрено выделение и классификация синтагматических зависимостей на основе множества СЭ-фраз. Предположим теперь, что элементами множества R в составе тройки (1.1) являются произвольные отношения между объектами $o \in O$. Кроме того, мы расширим возможности синонимического варьирования для ЕЯ-фраз множества Ts, введя синонимию на уровне предметной лексики наряду с лексико-функциональной.

Рассмотрим содержательные отличия процесса формирования множества R в составе тройки (1.1) при указанном расширении числа рассматриваемых случаев синонимии и произвольности отношений между объектами $o \in O$.

В задаче выделения и классификации синтаксических отношений в качестве основы формирования R относительно модели (1.1) мы брали множество неизменных частей всех слов, употребленных во всех фразах, представляемых множеством Ts. С учетом наличия РПЗ и конверсивов в словесном обозначении самой ситуации *S*, в роли слов, которые присутствуют во всех фразах синонимического множества, могли выступать только словесные обозначения "участников" ситуации.

Будем рассматривать введенное ранее индексное множество J применительно к неизменным частям всех слов, употребленных в более чем одной ЕЯ-фразе из множества Ts с учетом возможного неприсутствия слова во всех фразах указанного множества. При этом удвоенная длина общей неизменной части пары слов всегда больше суммы длин изменяемых (флективных) частей.

Последовательность индексов неизменных частей слов, присутствующих в $Ts_i \in Ts$, рассматривалась как модель линейной структуры этой фразы. Обозначим множество указанных моделей на J как LS. Тогда при наличии синонимов в словесных обозначениях либо участников ситуации S, либо характеристик участников будет справедливы следующие свойства моделей $Ls(Ts_i) \in LS$.

Теорема 5.1. Пара индексов $\{j_1, j_2\} \subset J$ соответствует словам-синонимам, если $\exists\{Ls(Ts_1), Ls(Ts_2)\} \subseteq LS : Ls(Ts_1) = J_1 \bullet \{j_1\} \bullet J_2$ и $Ls(Ts_2) = J_1 \bullet \{j_2\} \bullet J_2$, где $J_1 \subset J$, $J_2 \subset J$, а "$\bullet$" есть операция типа конкатенации над множеством *J*.

Доказательство теоремы следует из определения, сформулированного нами в разделе 3.5 для синтаксической связи применительно к модели линейной структуры предложения.

Пусть PJ – множество пар, отвечающих условию *теоремы 5.1*. Заменим индексы, вошедшие в пары из PJ, на некоторые $j \in (\mathrm{N} \setminus J)$ во всех моделях из LS, где N – множество натуральных чисел. Обозначим преобразованное LS как LS', множество заменяемых индексов – как JP, а множество индексов, на кото-

рые производится замена, – как JP', $JP' \cap JP = \varnothing$. Фактически каждая модель в LS' задается на множестве $(J \setminus JP) \cup JP'$.

Утверждение 5.1. Справедливым будет утверждать, что индексы с максимальной встречаемостью в различных моделях из множества LS' соответствуют словам-существительным, обозначающим участников ситуации (1.1).

Доказательство утверждения следует из доказанной *теоремы 5.1* и сделанного допущения о наличии РПЗ и конверсивов в словесных обозначениях ситуаций.

Обозначим множество индексов, удовлетворяющих *утверждению 5.1*, как JN. Пусть $Ls_1(Ts_i) \in LS'$, а $Ls_2(Ts_i)$ – модель линейной структуры той же фразы Ts_i, но относительно JN. Обозначим множество моделей второго вида как LJN. Положим также, что имеется $LS'_j \subset LS'$ такое, что для всех $Ls_1(Ts_i) \in LS'_j$ модели $Ls_2(Ts_i)$ одинаковы и соответствуют некоторой $Ls_2(Ts_j) \in LJN$, $Ts_j \in Ts$.

Утверждение 5.2. Индексы $j \notin JN$ с максимальной частотой встречаемости в различных моделях $Ls_1(Ts_i) \in LS'_j$ соответствуют либо словам-наречиям, либо прилагательным, либо опорным существительным в составе генитивных конструкций.

Доказательство. Исключением из множества LS'_j тех моделей, все индексы в составе которых входят в JN, с последующим удалением индексов $j \in JN$ из оставшихся моделей, получаем частный случай *утверждения 5.1*.

Обозначим множество индексов, удовлетворяющих *утверждению 5.2*, как JA. Установление синтаксических ролей и выделение флексий для слов с индексами из $((J \setminus JP) \cup JP') \setminus (JN \cup JA) \cup \{0\}$ производится по аналогии с выявлением указанной информации у слов в составе РПЗ способом, описанным в разделе 3.5. При этом вместо индексов с ненулевым значением рассматриваются индексы из $JN \cup JA$.

Таким образом, в соответствии с требованием иерархичности знаний о синонимии множество R отражает:

– сочетаемость основ синтаксически главных и зависимых слов. Данный вид отношений необходим для выделения объектов и признаков во всех рассматриваемых видах синонимии;

– сочетаемость флексий главных и зависимых слов. Фактически здесь задаются значения признаков для классов СЭ;

– сочетаемость слова и его лексико-семантических производных в рамках РПЗ. Указанные отношения значимы для выделения и классификации случаев лексико-функциональной синонимии.

Сами семантические отношения при этом составляют основу классификации и вычисления оценки схожести ситуаций употребления ЕЯ.

5.2. Формальный контекст ситуации языкового употребления и методы его построения

Задача классификации и анализа схожести ситуаций употребления ЕЯ наиболее естественно решается методами АФП, рассмотренными в предыдущих главах.

Отметим особенности объектов и признаков для отдельной ситуации языкового употребления, представляемой тройкой вида (1.1), и для совокупности таких ситуаций, подлежащих сравнению.

Множество объектов Gs формального контекста

$$Ks = (Gs, Ms, Is) \tag{5.1}$$

одной ситуации составляют основы слов, входящих во фразы из множества Ts и являющихся зависимыми по отношению к другому слову из некоторой ЕЯ-фразы $Ts_i \in Ts$.

Множество признаков Ms включает в себя подмножества, обозначаемые далее посредством соответствующего нижнего индекса и содержащие:

– указания на основу синтаксически главного слова (*индекс 1*);

– указания на флексию главного слова (*индекс 2*);

– связи “основа – флексия” для синтаксически главного слова (*индекс 3*);

– сочетания флексий зависимого и главного слова (*индекс 4*). При этом после флексии главного слова через двоеточие указывается предлог (если такой имеется) для связи главного слова с зависимым;

– указания на флексию зависимого слова (*индекс 5*).

Посредством $Is \subseteq Gs \times Ms$ отношения из множества R разбиваются на классы по сходству:

– основы главного слова, что особенно актуально для исследования сочетаемости в рамках ЛФ-параметров, посредством которых описываются РПЗ;

– флексии зависимого слова, что необходимо для выделения и обобщения синтаксических отношений;

– лексической и флективной сочетаемости, что позволяет выявить зависимости, аналогичные смысловой связи между опорным словом и генитивной именной группой в составе генитивной конструкции русского языка.

При этом каждому классу соответствует некоторое формальное понятие в решетке $\mathfrak{Rs}(Gs, Ms, Is)$.

Решетка $\mathfrak{Rs}$ для примера ситуации ЕЯ-употребления, рассмотренного в разделе 3.5, представлена на рис. 5.1. Здесь ранее использованное СЭ-множество дополнено новыми ЕЯ-фразами, полученными из уже имеющихся фраз путем синонимических замен как абстрактных слов и их сочетаний (“*является следствием*” – “*служит причиной*”), так и предметной лексики (“*переобучение*” – “*переподгонка*”). В целях компактности изложения графического материала в формальный контекст не были включены объекты и признаки для прилагательных (“*эмпирический*” и “*нежелательное(ая)*”).

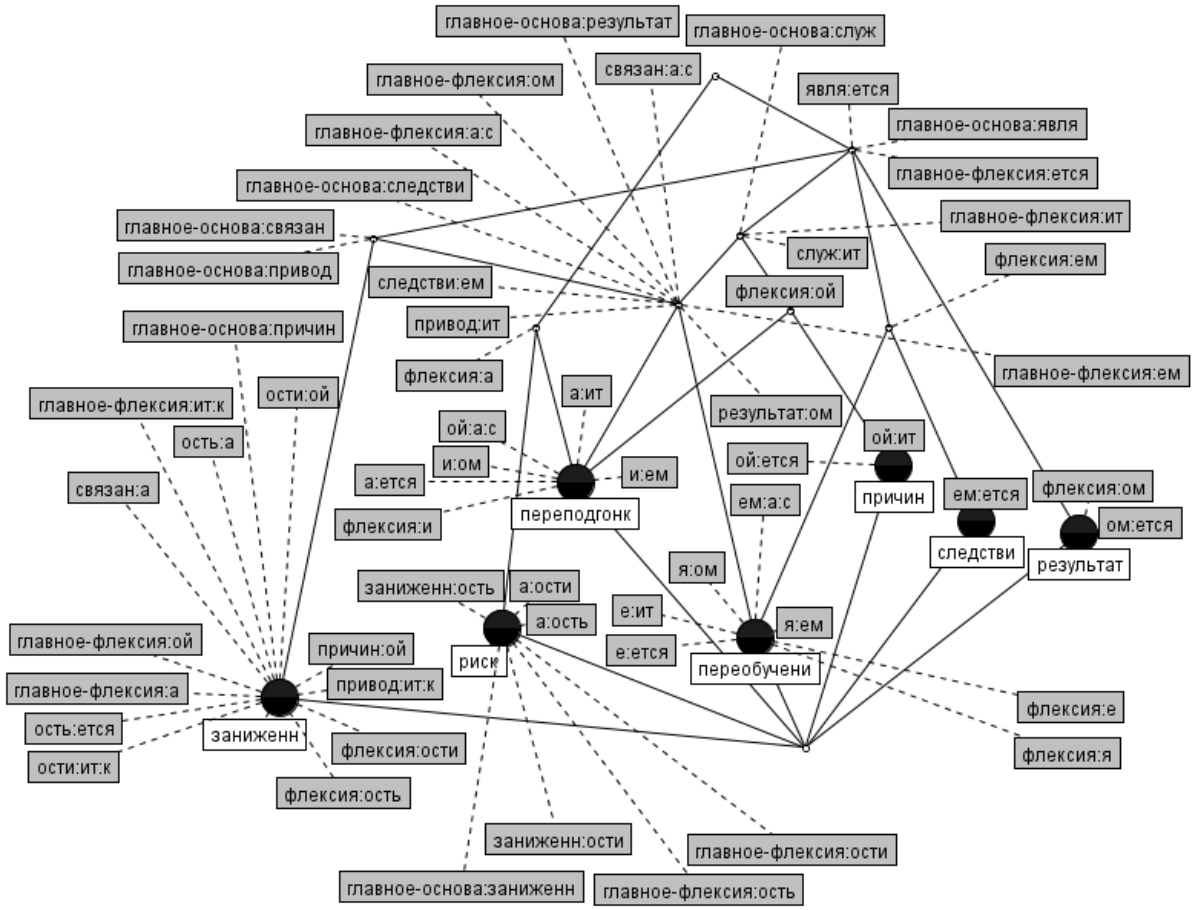


Рис. 5.1. Пример формального контекста ситуации языкового употребления

Классы формальных понятий в решетке различаются степенью абстракции, которая зависит от частоты употребления главных слов анализируемых сочетаний в различных синтаксических контекстах относительно модели (1.1). Для количественной оценки СЭ значимы классы одного уровня абстракции, соответствующие подчинению существительных, обозначающих участников ситуации, тем словам, которые ее называют и не входят в РПЗ. *Необходима* редукция контекста вида (5.1) исключением объектов и признаков РПЗ.

Утверждение 5.3. Пусть $\{m_1, m_2, m_3\} \subset M_1$. Если считать m_1, m_2 и m_3 взаимно различными, то m_1 соответствует указанию на основу главного, m_2 – зависимого слова РПЗ, а m_3 – указанию на основу однословного эквивалента РПЗ при выполнении трех условий:

1. $\exists g_1 \in Gs : Is(g_1, m_1) = true$, $Is(g_1, m_3) = false$, $m_2 = p_{bs} \bullet g_1$. Здесь символ "$\bullet$" обозначает конкатенацию, а p_{bs} есть используемое далее обозначение для символьной константы *"главное – основа:"*.

2. $\exists \{g_2, g_3\} \subset Gs$, при этом объекты g_1, g_2 и g_3 взаимно различаются, а

$$Is(g_2, m_3) \wedge Is(g_3, m_3) \wedge$$

$$\wedge (Is(g_2, m_1) \wedge Is(g_3, m_2) \vee Is(g_2, m_2) \wedge Is(g_3, m_1)) = true.$$

3. Не существует других троек объектов, для которых признак m_3 занимал бы место либо признака m_1, либо признака m_2 в вышеуказанных соотношениях.

Доказательство утверждения следует из свойств базиса импликаций для формального контекста вида (5.1).

Исключая объекты и признаки слов расщепленных предикатных значений согласно *утверждению 5.3* для приведенного на рис. 5.1 примера, получаем редуцированный формальный контекст, решетка ФП для которого представлена на рис. 5.2.

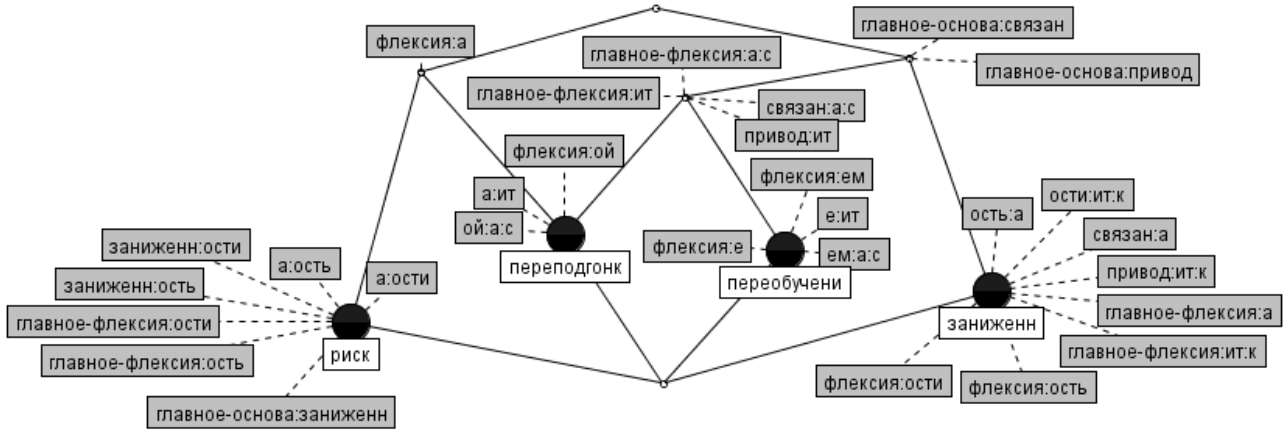


Рис. 5.2. Решетка ФП для редуцированного формального контекста

После удаления информации РПЗ формальный контекст вида (5.1) отражает классы отношений, которые определяются исключительно ролями объектов – участников ситуации по отношению к ней самой. При этом синтаксические зависимости как частный случай семантических отношений выражаются определен-

ными сочетаниями флексий. Сказанное позволяет в ряде случаев выделять основы и их сочетания на базе указанных морфологических зависимостей. Эти зависимости могут быть либо выявлены ранее для других ситуаций языкового употребления, либо найдены с помощью программ синтаксического анализа, реализующих стратегию разбора на основе наиболее вероятных связей слов. Фактически данные связи и выделяются согласно принципу, который был предложен в разделе 3.5 и дополнен в настоящей главе.

5.3. Тезаурус предметной области и схожесть ситуаций языкового употребления

Рассмотрим теперь *задачу* накопления и систематизации знаний, представляемых структурами вида (5.1). Если указанные знания формируются на основе независимого ЕЯ-описания различных фактов некоторой предметной области группой экспертов, то получаемая структура будет соответствовать тезаурусу этой предметной области. При этом предполагается, что: а) из множеств объектов и признаков каждой рассматриваемой ситуации языкового употребления удалена информация расщепленных предикатных значений; б) выделение самих объектов и признаков производится как на основе принципа, предложенного в настоящей работе, так и с помощью известных синтаксических анализаторов.

Заметим, что число форм языкового описания для модели (1.1) изначально не оговаривается. Фактически это означает то, что слова, являющиеся синонимами по *теореме 5.1*, могут обозначать понятия с различной степенью абстракции. На практике указанная степень тем больше, чем больше число ситуаций, представляемых тройками вида (1.1), относительно которых понятие фигурирует в некоторой фиксированной роли.

Возьмем указанный факт за основу численной оценки схожести для ситуаций языкового употребления, порождаемых независимо друг от друга.

Представим тезаурус, формируемый на основе совокупности ситуаций ЕЯ-употребления для известных фактов заданной предметной области, посредством формального контекста:

$$Kth = (Gth, Mth, Ith). \qquad (5.2)$$

При этом множество объектов Gth составляют символьные пометки, присваиваемые отдельным ситуациям. Множество Mth включает элементы множеств признаков формальных контекстов вида (5.1) всех $gth \in Gth$. Кроме того, в составе Mth выделяются:

– множество указаний на объекты формальных контекстов (5.1), генерируемых для элементов множества Gth (обозначим далее это множество как M_6);

– множество связей “основа – флексия” для зависимого слова (M_7);

– множество сочетаний основ зависимого и главного слова (M_8).

На рис. 5.3 формальный контекст из примера на рис. 5.2 представлен одним формальным понятием для объекта $gth \in Gth$.

Другие факты этой же предметной области “Математические методы обучения по прецедентам” [6, 7, 9, 23–27, 36, 37, 40, 157], использованные для генерации тезауруса, приведены в табл. 5.1. Представление тезауруса решеткой формальных понятий приведено на рис. 5.4.

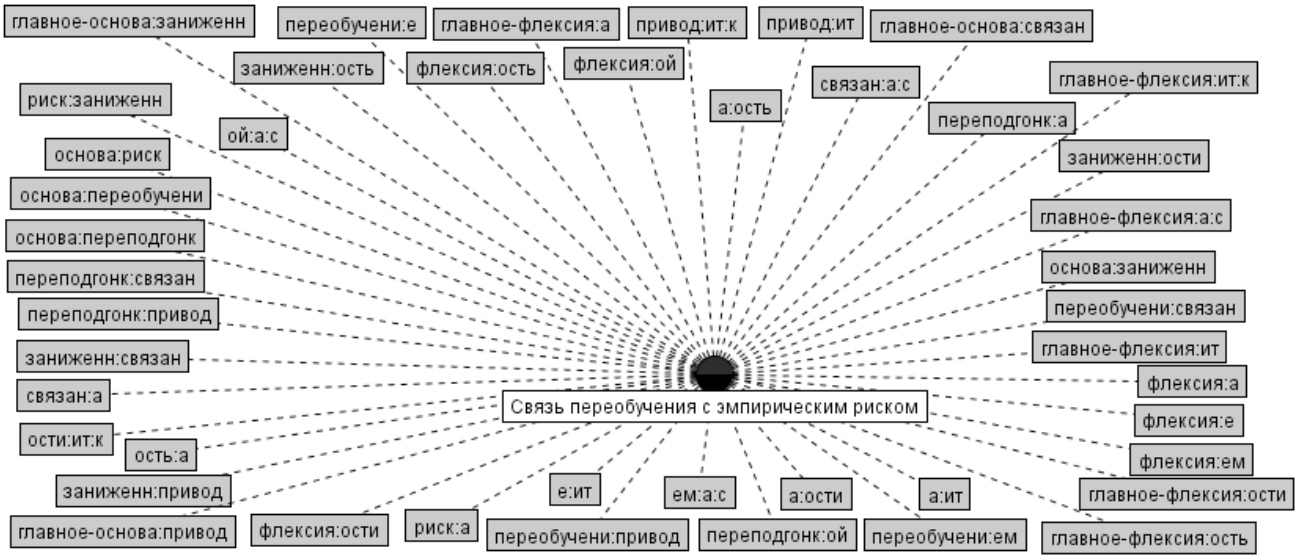


Рис. 5.3. Ситуация ЕЯ-употребления как объект формального контекста тезауруса

Таблица 5.1

Исходные данные для построения тезауруса

№ п/п	1				2	3		4	
Основа	Флективная часть + предлог								
заниженн	ость	ость	ости	ости	–	ость	ости	ость	ость
оценк	–	–	–	–	–	и	и	и	и
эмпирическ	ого	–	ого	–	–	–	–	–	–
риск	а	–	а	–	–	–	–	–	–
средн	–	ей	–	ей	–	–	–	–	–
ошибк	–	и:на	–	и:на	–	–	–	и	и
распознавани	–	–	–	–	–	–	–	я	я
обучающ	–	ей	–	ей	–	–	–	–	–
выборк	–	е	–	е	–	–	–	–	–
переусложнени	ем	ем	е	е	–	–	–	–	–
модел	и	и	и	и	–	–	–	–	–
уменьшени	–	–	–	–	е	–	–	–	–
обобщающ	–	–	–	–	ей	ей	ей	–	–
способност	–	–	–	–	и	и	и	–	–
выбор	–	–	–	–	–	–	–	ом	а
решающ	–	–	–	–	его	–	–	его	его
дерев	–	–	–	–	а	–	–	–	–
правил	–	–	–	–	–	–	–	а	а
алгоритм	–	–	–	–	–	а	а	–	–
переподгонк	–	–	–	–	ой	ой	а	–	–
переобучени	–	–	–	–	–	ем	е	–	–
связан	а:с	а:с	–	–	о:с	а:с	–	а:с	–
вызван	а	а	–	–	–	а	–	–	–
обусловлен	а	а	–	–	о	–	–	–	–
привод	–	–	ит:к	ит:к	–	–	ит:к	–	–
завис	–	–	–	–	–	–	–	–	ит:от

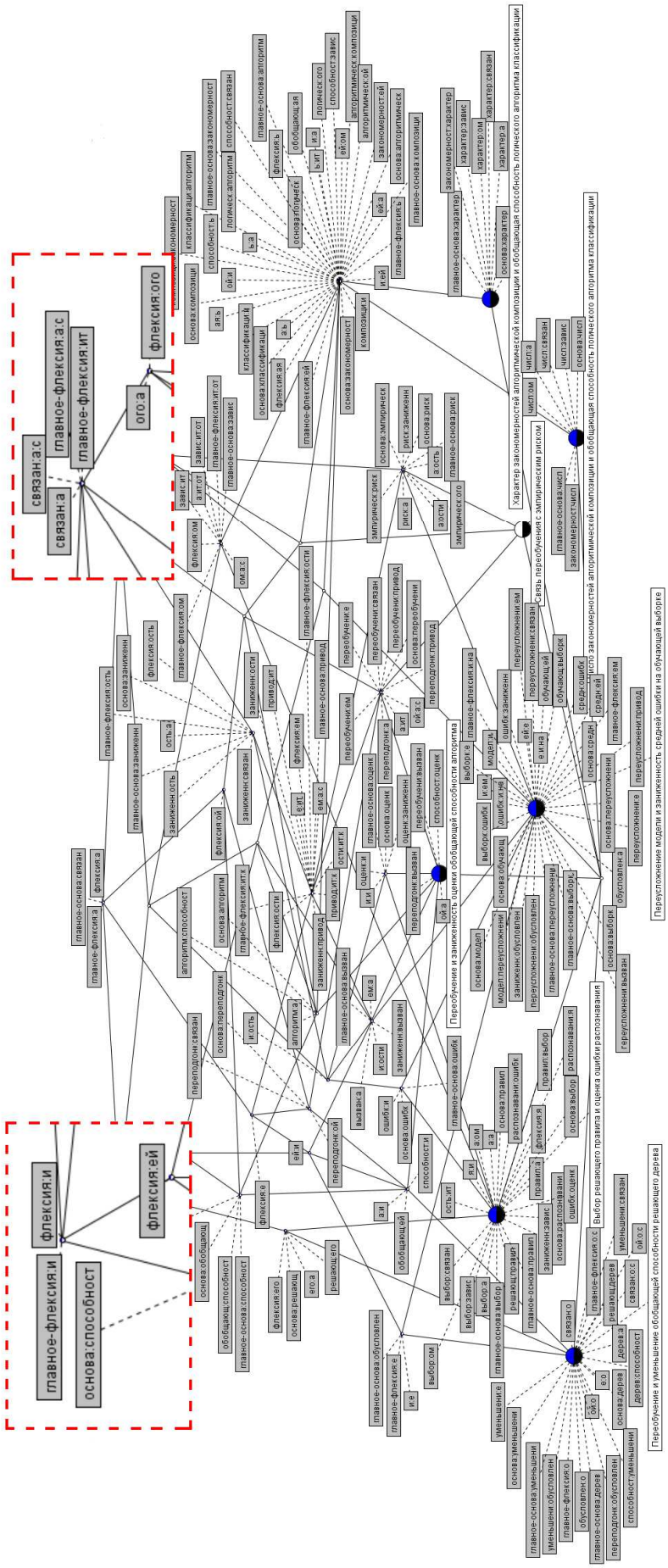


Рис. 5.4. Решетка ФП тезауруса и классы синтаксических отношений

Пусть S_1 – ситуация вида (1.1), соответствующая заведомо корректному ("эталонному") ЕЯ-описанию некоторого известного факта заданной предметной области. Положим также, что S_2 – анализируемая ситуация, для которой соответствие ситуации S_1 и имеющимся предметным знаниям заранее неизвестно. Обозначим используемые в дальнейших рассуждениях формальные контексты вида (5.1): для ситуации S_1 – как Ke, а для ситуации S_2 – как Kx, где $Ke = (Ge, Me, Ie)$ и $Kx = (Gx, Mx, Ix)$, $Ie \subseteq Ge \times Me$ и $Ix \subseteq Gx \times Mx$, соответственно. Введем также обозначения для используемых далее символьных констант: p_{fl} – для "флексия:", p_b – для "основа:". В соответствии с показанным выше разделением множества признаков формального контекста вида (5.1) будем обозначать соответствующие подмножества в составе Me и Mx как Me_k и Mx_k, где $k = 1, \ldots, 5$. Результат объединения множеств M_6, M_7, M_8, Me_4, Mx_4, Me_5 и Mx_5, обозначим как M_U.

Определение 5.1. Будем считать, что S_1 и S_2 связаны отношением схожести, если каждому объекту $gx \in Gx$ соответствует такой объект $ge \in Ge$, что выполняется одно из следующих условий:

(1) $gx = ge$ и любой признак $me \in Me$ объекта ge относится и к gx.

(2) $gx = ge$, при этом *условие (1)* не выполняется, но существует $gth \in Gth$, обладающий признаком $mth_1 \in M_6$: $mth_1 = p_b \bullet ge$ при обязательном выполнении следующих условий:

$(\exists\, me_{fl} \in Me_5 : me_{fl} = p_{fl} \bullet fe) \rightarrow (\exists\, mth_{17} \in M_7 : mth_{17} = ge \bullet ":" \bullet fe)$,

при этом $(Ie(ge, me_{fl}) \wedge Ix(ge, me_{fl})) \rightarrow Ith(gth, mth_{17})$;

$(\exists\, me_{bs} \in Me_1 : me_{bs} = p_{bs} \bullet be) \rightarrow (\exists\, mth_{18} \in M_8 : mth_{18} = ge \bullet ":" \bullet be)$,

при этом $Ie(ge, me_{bs}) \rightarrow Ith(gth, mth_{18})$;

$(\exists\, mx_{bs} \in Mx_1 : mx_{bs} = p_{bs} \bullet bx) \rightarrow (\exists\, mth_{28} \in M_8 : mth_{28} = ge \bullet ":" \bullet bx)$,

при этом $Ix(ge, mx_{bs}) \rightarrow Ith(gth, mth_{28})$.

Кроме того, для $\forall mth \in (Mth \setminus M_U)$ истинно:

$$Ith(gth, mth) \rightarrow (Ie(ge, mth) \wedge Ix(ge, mth)). \quad (5.3)$$

В содержательном плане *условие (2)* настоящего *определения* описывает случай наличия синонимов среди слов, синтаксически главных по отношению к словам со сходными основами. При этом основы gx и ge не омонимичны, поскольку в этом случае было бы нарушено требование разделения ими признаков главного слова.

(3) $gx \neq ge$, но существует объект $gth \in Gth$, обладающий признаками $mth_1 \in M_6 : mth_1 = p_b \bullet ge$ и $mth_2 \in M_6 : mth_2 = p_b \bullet gx$, при этом для любого признака $mth \in (Mth \setminus M_U)$ справедливо:

$$Ith(gth, mth) \rightarrow (Ie(ge, mth) \wedge Ix(gx, mth)). \quad (5.4)$$

(4) $gx \neq ge$, но существует объект $gth_1 \in Gth$, обладающий признаком $mth_1 \in M_6 : mth_1 = p_b \bullet ge$, а для $\forall me \in (Me_4 \cup Me_5)$ верно:

$$(Ith(gth_1, mth_1) \wedge Ie(ge, me)) \rightarrow Ith(gth_1, me).$$

При этом существуют признаки $mth_2 \in M_6 : mth_2 = p_b \bullet gxg$ и $mx \in (Mx_1 \cup Mx_2 \cup Mx_3)$, для которых верно:

$$(Ith(gth_1, mth_2) \wedge Ix(gx, mx)) \rightarrow Ith(gth_1, mx),$$

где $gxg \neq gx$, а пара (gxg, ge) отвечает *условию (3)* при генерации формального контекста вида (5.1) для объекта gth_1. В то же время существует объект $gth_2 \in Gth$, относительно которого пара (gx, gxg) также будет отвечать *условию (3)* настоящего *определения*. Генерируемый при этом ФК вида (5.1) для gth_2 обозначим как Kxg, $Kxg = (Gxg, Mxg, Ixg)$.

Замечание. Численная оценка схожести ситуаций S_1 и S_2 включает сравнение последовательностей двух и более соподчиненных слов. Пример (см. табл. 5.1 и рис. 5.4): *“средняя ошибка на обучающей выборке”* ⇔ *“эмпирический риск”*. Выполнимость условий *определения 5.1* здесь анализируется только для

главных слов (в примере это *“ошибка”* и *“риск”*). Сами последовательности считаются взаимно заменяемыми, если возможно их построение по формальному контексту (5.2) на наборе признаков с префиксом p_{bs} для одной и той же ситуации языкового употребления. При этом главные слова последовательностей должны быть одинаково подчинены одному и тому же слову, что проверяется по сочетанию флексий.

Таким образом, *определение 5.1* учитывает уровень абстракции понятий, обозначаемых словами с основами gx и ge, при сходстве их синтаксических ролей, определяемых признаками из множеств Me_4, Mx_4, Me_5 и Mx_5. При этом само синтаксическое отношение выступает своего рода обобщением ряда семантических отношений. Это подтверждается, в частности, анализом классов ФП в решетке, генерируемой на основе ЕЯ-описаний известных фактов предметной области: отношениям, определяемым сочетаниями флексий, как правило, соответствуют классы более высокого уровня абстракции (в примере на рис. 5.4 эти классы выделены прямоугольниками). Сказанное позволяет в целом провести аналогию между схожестью формальных понятий в рамках одного контекста и схожестью самих формальных контекстов. Этому вопросу посвящен следующий раздел.

5.4. Интерпретация меры схожести формальных понятий для формальных контекстов

Понятие схожести между языковыми контекстами, определяемыми структурами вида (1.1), определяется индуктивно на основе представления о семантическом расстоянии между отдельными лексемами, обсуждавшегося в [120].

Действительно, семантическая схожесть как разновидность семантического расстояния основана на отношении порядка, которое включает родовидовое отношение, отношение синонимии, отношение сочинения и отношение атрибуции между объектами и признаками в формальном контексте. А поскольку только отношение порядка может быть извлечено из решетки формальных понятий,

то именно данный вид отношений и должен служить основой схожести между языковыми контекстами.

Согласно данному в [120] определению, полная синонимия между словами с основами $\{g_1, g_2\}$ будет иметь место тогда, когда объекты g_1 и g_2 принадлежат объему одного и того же понятия формального контекста (5.1) некоторой ситуации языкового употребления. Фактически именно этот случай и обобщается *условием (1) определения 5.1* уже на взаимно различные формальные контексты. Отношение сочинения, как показано в [120], существует между объектами формальных понятий с одинаковым НОСП. Частные случаи такого отношения для объектов из взаимно различных формальных контекстов описывается *условиями (2)* и *(3) определения 5.1*.

Более сложные случаи отношения порядка на основе композиции сочинения и родовидового отношения (гипонимии) рекурсивно определяет *условие (4) определения 5.1*. Как следует из данного условия, и для взаимно различных формальных контекстов схожесть объектов тем больше, чем более специфичным является их НОСП.

Таким образом, основой численной оценки схожести формальных контекстов должна быть общая информация, разделяемая объектами из разных контекстов, а также специфичность общей информации, вычисляемой по расстоянию от вершины в иерархии контекстов, которая в рассматриваемой нами задаче представляется решеткой для формального контекста вида (5.2).

Обобщая *определение 5.1*, будем считать, что формальные контексты связаны отношением схожести, если каждому ФП одного контекста можно поставить в соответствие такое ФП второго контекста, что при этом между формальными понятиями становится возможным установление отношения порядка.

Для введения численной оценки схожести между формальными контекстами рассмотрим обобщенный способ прочтения формул (5.1) и (5.2).

Множество *Gth* в (5.2) составляют символьные пометки, присваиваемые отдельным контекстам вида (5.1). Объединение множеств M_7 и M_8 в общем

случае получает содержательную интерпретацию множества связей между признаками из множества Mth, каждая из которых соответствует некоторой связи объекта и признака конкретного формального контекста $gth \in Gth$, представленного в форме (5.1). Таким образом, на основе совокупности троек вида (5.1) и (5.2) могут быть рекурсивно определены многоуровневые формальные контексты по аналогии с сетями Петри высокого уровня [59], характерный пример которых был фактически рассмотрен нами во второй главе. Мера схожести формальных понятий из контекстов одного уровня рекурсивного вложения определяется аналогично схожести формальных понятий внутри одного контекста. При этом для применения соотношений, описанных в [120], объекты и признаки пары сравниваемых формальных контекстов вида (5.1) должны быть трансформированы в признаки формального контекста вида (5.2), множество типа M_6 для которого содержит указания на объекты обоих формальных контекстов из указанной пары. При установлении степени схожести ситуаций языкового употребления число вышеуказанных уровней рекурсивного вложения равно двум: нижний уровень представлен формальными контекстами сравниваемых ситуаций, верхний – тезаурусом предметной области.

5.5. Смысловая близость фраз предметно-ограниченного подмножества естественного языка

Рассмотрим применение формализованного представления тезауруса вида (5.2) для численной оценки схожести ситуаций языкового употребления, представляемых формальными контекстами вида (5.1). За основу возьмем предложенную в [120] меру схожести для формальных понятий в пределах одной решетки.

С учетом выполняемого в соответствии с *определением 5.1* сопоставления объектов формальных контекстов $Ke = (Ge, Me, Ie)$ и $Kx = (Gx, Mx, Ix)$, из которых удалена информация РПЗ, схожесть ситуаций S_1 и S_2 может быть численно оценена как

$$spc(S_1, S_2) = \frac{\sum_{k=1}^{n} spc_k}{n}, \tag{5.5}$$

где $n = |Gx|$, а spc_k есть численное значение схожести объектов в паре (gx_k, ge). В зависимости от выполнимости условий *определения 5.1* значение spc_k :

– равно 1,0, если для пары (gx_k, ge) выполнено *условие (1)*;

– вычисляется по формуле:

$$-\log_2\left(1 - \frac{D_c}{path_C}\right) \times \frac{|BLCS|}{|B_1 \setminus BLCS| + |B_2 \setminus BLCS| + |BLCS|}, \tag{5.6}$$

если для пары (gx_k, ge) выполнено *условие (2), (3)* либо *(4)*.

Во втором случае мы имеем дело с гипотетической решеткой ФП (обозначим ее как $\Re xe$), в которой объемы объектных формальных понятий (формальных понятий с одним объектом в составе объема) есть $\{gx_k\}$ и $\{ge\}$ (при выполнении *условия (2)* или *(3)*) либо $\{gx_k\}$, $\{ge\}$ и $\{gxg\}$ (при выполнении *условия (4)*). Значение D_c равно числу сравнимых формальных понятий, составляющих цепочку с вершинным ФП решетки $\Re xe$ в качестве максимального ФП и наименьшим общим суперпонятием для объектных формальных понятий решетки $\Re xe$ – в качестве минимального ФП. Множество $BLCS$ есть содержание этого НОСП, а число $path_C$ равно минимальному числу формальных понятий в цепочке, которой принадлежит вершинное ФП, наименьшее ФП решетки $\Re xe$ и формальное понятие с содержанием $BLCS$.

В случае выполнения любого из *условий (2), (3)* или *(4)* значение $D_c = 2$ (доказательство очевидно). При выполнении *условия (2)* либо *(3)* число $path_C = 4$, а в множество $BLCS$ войдут признаки $mth \in (Mth \setminus M_U)$, для каждого из которых справедливо либо соотношение (5.3) (при выполнении *условия (2)*), либо соотношение (5.4) (при выполнении *условия (3)*). Множества B_1 и B_2 в этом случае определяются следующим образом:

$$B_1 = \{\, me : me \in (Me_1 \cup Me_2 \cup Me_3),\ \ Ie(ge, me) = true\},$$

$$B_2 = \{\, mx : mx \in (Mx_1 \cup Mx_2 \cup Mx_3),\ \ Ix(gx_k, mx) = true\}.$$

Доказательство выполнимости *условия (4)* обычно происходит в несколько итераций. При этом в ходе каждой последующей итерации число признаков, не являющихся общими для gx_k и gxg, всегда меньше, чем в предыдущей. Начальное значение числа $path_C$, равное 4, в ходе каждой итерации увеличивается на 1, а

$$B_1 = \{mxg : mxg \in (Mxg_1 \cup Mxg_2 \cup Mxg_3),\ \ Ixg(gxg, mxg) = true\},$$

$$B_2 = \{mx : mx \in (Mxg_1 \cup Mxg_2 \cup Mxg_3),\ \ Ixg(gx_k, mx) = true\},$$

где $(Mxg_1 \cup Mxg_2 \cup Mxg_3) \subset Mxg$ согласно показанному выше разделению множества признаков формального контекста вида (5.1), а $BLCS$ в этом случае есть пересечение множеств B_1 и B_2.

Значения $|B_1|$ и $|B_2|$ в формуле (5.6) будут тем больше, чем большее число слов могут быть синтаксически главными по отношению к каждому из слов для пары (gx_k, ge). При этом величина $|BLCS|$ отражает взаимную специфичность понятий, обозначаемых gx_k и ge.

В качестве примера рассмотрим ЕЯ-описание факта наличия связи между переобучением и эмпирическим риском, представленное для ситуации S_1 четырьмя синонимичными простыми распространенными предложениями русского языка. Положим, что S_1 соответствует сформулированному разработчиком теста варианту правильного ответа на тестовое задание открытой формы.

Предложения 1 и 2: *“Переобучение (=переподгонка) приводит к заниженности эмпирического риска”*. Предложения 3 и 4: *“Заниженность эмпирического риска связана с переподгонкой (=переобучением)”*.

Выполнив синтаксический разбор программой “Cognitive Dwarf”, выделяем основы, флексии и их сочетания. Получаем формальный контекст вида (5.1), представленный решеткой формальных понятий на рис. 5.5.

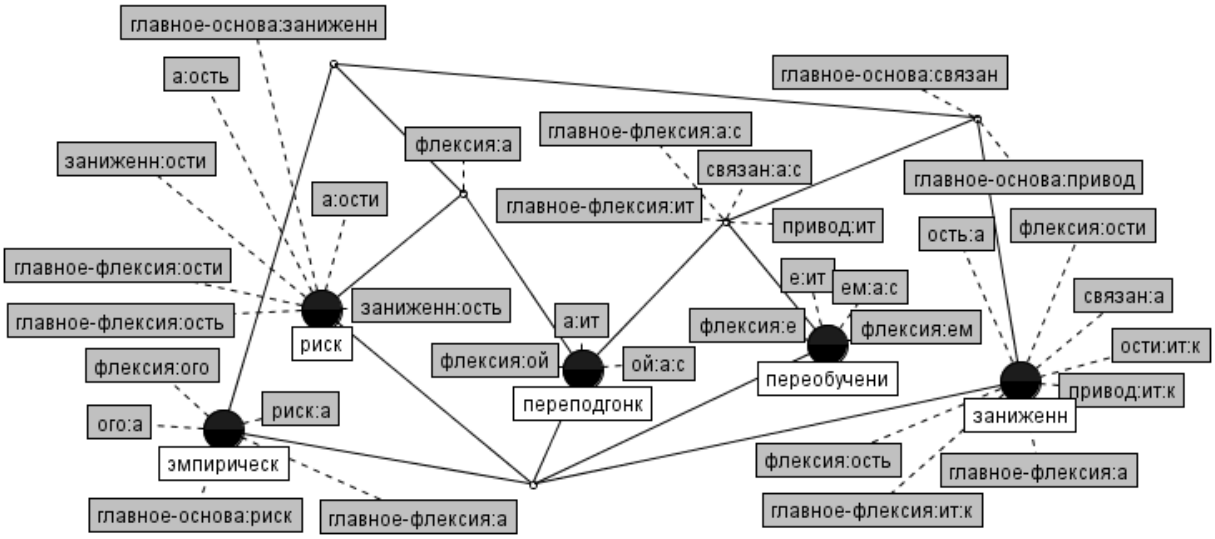


Рис. 5.5. Ситуация ЕЯ-употребления для “эталонного” описания заданного факта

Теперь предположим, что мы имеем три анализируемых независимых варианта ЕЯ-описания ситуации S_2, причем все три связаны отношением схожести с ситуацией S_1 согласно *определению 5.1*. Каждый из них описывает тот же факт, что и S_1 – наличие связи между переобучением и эмпирическим риском, причем описание выполнено одним простым распространенным предложением русского языка. Положим, что указанные варианты соответствуют ответам трёх испытуемых на тестовое задание открытой формы, правильный ответ на которое отождествляется с S_1.

Первый вариант: *“Заниженность средней ошибки на обучающей выборке связана с переобучением”*. Второй вариант: *“Заниженность средней ошибки на обучающей выборке связана с переподгонкой”*. Третий вариант: *“Переобучение приводит к заниженности средней ошибки на обучающей выборке”*.

Как и для ситуации S_1, формальные контексты вида (5.1) здесь строятся на основе результатов синтаксического разбора предложений программой “Cognitive Dwarf”. Полученные решетки формальных понятий представлены на рис. 5.6, 5.7 и 5.8.

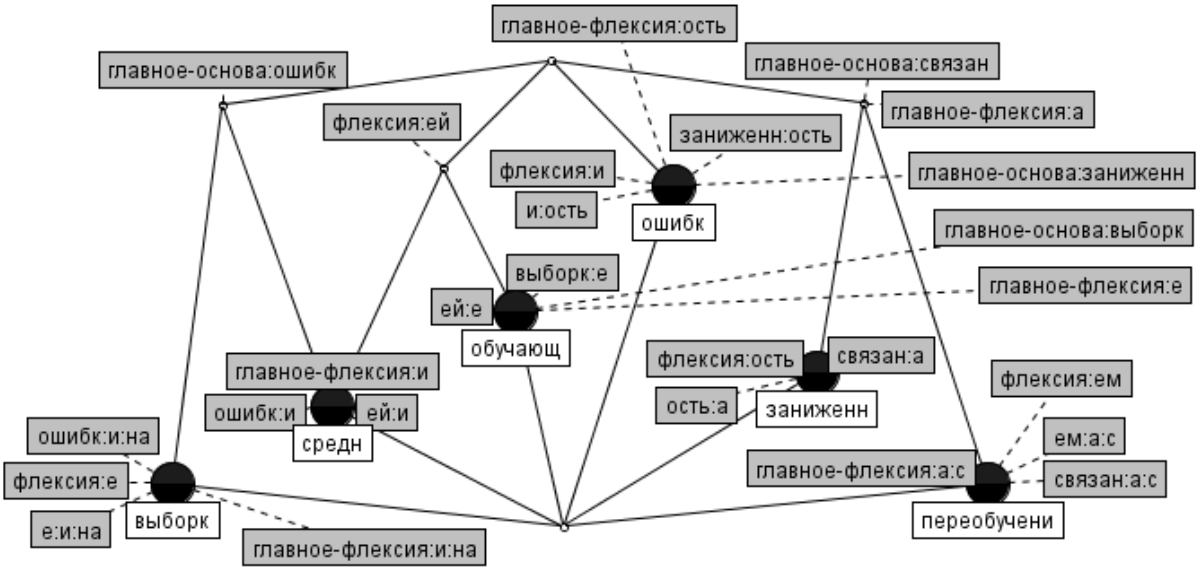


Рис. 5.6. Вариант 1 анализируемого ЕЯ-описания связи переобучения с эмпирическим риском

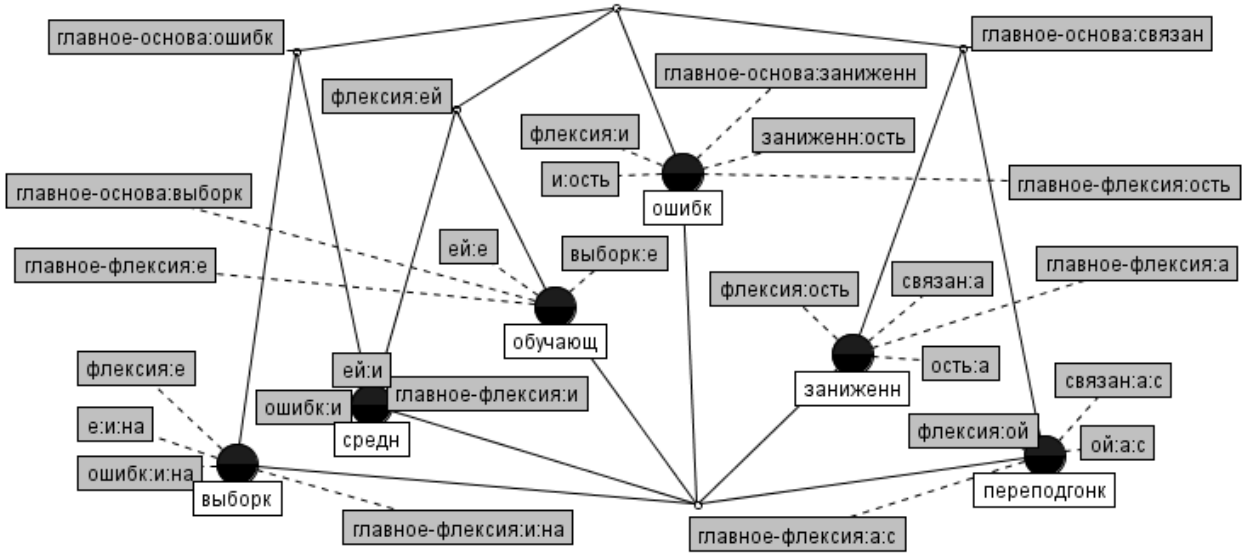


Рис. 5.7. Вариант 2 анализируемого ЕЯ-описания связи переобучения с эмпирическим риском

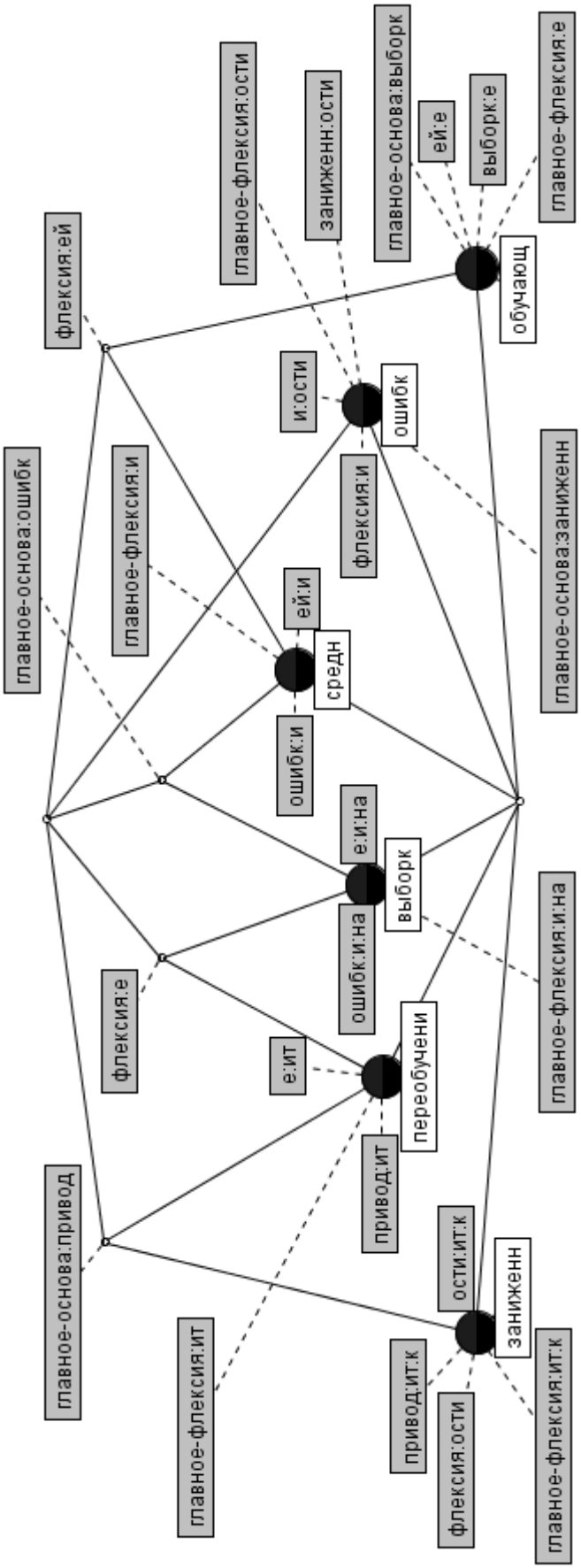


Рис. 5.8. Вариант 3 анализируемого ЕЯ-описания связи переобучения с эмпирическим риском

Таблица 5.2

Сравнение вариантов ЕЯ-описания ситуации S_2

Вариант	$spc(S_1, S_2)$	$\lvert BLCS \rvert$	$\lvert B_1 \setminus BLCS \rvert$	$\lvert B_2 \setminus BLCS \rvert$
1	0,9167	7,7500	0,7500	0,0000
2	0,7917	7,0000	2,0000	0,5000
3	0,8750	7,7500	0,7500	0,7500

Как видно из табл. 5.2, наибольшее значение схожести с ситуацией S_1 по формуле (5.5) имеет *вариант 1* ЕЯ-описания ситуации S_2.

Действительно, для этого варианта в формуле (5.6) мы имеем наибольшее среднее значение $\lvert BLCS \rvert$ при минимальном среднем значении суммы $\lvert B_1 \setminus BLCS \rvert$ и $\lvert B_2 \setminus BLCS \rvert$ по всем парам (gx_k, ge), для которых выполняется *условие (2), (3)* либо *(4) определения 5.1*. Причина состоит в том, что признаки объектов формального контекста, соответствующего *варианту 1*, разделяются большим числом объектов формального контекста ситуации S_1, чем признаки у объектов формальных контекстов для *вариантов 2* и *3*. Иными словами, признаки для *варианта 1* являются более стереотипическими по отношению к формальному контексту ситуации S_1, чем признаки у двух других вариантов.

Немаловажную роль при вычислении оценки схожести ситуаций языкового употребления играет также полнота и непротиворечивость ЕЯ-описания предметных знаний при формировании тезауруса. Предложенная модель тезауруса в виде решетки формальных понятий позволяет задействовать, в частности, базис импликаций формального контекста (5.2) для изучения взаимозаменяемости абстрактных слов в синтаксических контекстах существительных предметной лексики (“*связана с переобучением*” ⇔ “*переобучение приводит (к)*”). Соотнесение соответствующих классов ФП решетки тезауруса с уже известными классами семантической эквивалентности в заданном ЕЯ – тема отдельного рассмотрения.

5.6. Сжатие текстовой информации на основе теоретико-решеточного подхода: проблемы и перспективы

В настоящем разделе мы вкратце остановимся на основных вопросах использования модели (5.2) в качестве основы построения текстовых баз данных для заданной предметной области. Сразу отметим, что разработка полной архитектуры СУБД на основе теоретико-решеточного подхода не является предметом рассмотрения автора в настоящей работе.

Во-первых, для организации самой базы данных в рамках любой из известных на сегодняшний день моделей необходимо определиться с набором отношений, непосредственно определяющих данные. В качестве такого набора вполне может выступать совокупность характеристических функций, определяющих смысл текста. Данное определение естественным образом вытекает из формального определения смысла слова, сформулированного в главе 3, и на основе рассуждений, проделанных в главах 4 и 5 относительно синтаксического контекста имени существительного.

Во-вторых, при использовании смысла как набора атрибутов текста актуальна проблема избыточности данных, в первую очередь вызванная наличием расщепленных предикатных значений. Согласно общеизвестным правилам нормализации отношений [20, с. 397–504], связи между главным и зависимым словом в составе РПЗ, а также между РПЗ и его нерасщепленным эквивалентом, должны быть представлены отдельно от связей между участниками ситуаций и самими ситуациями. Модель (5.2) решает указанную задачу даже если из формальных контекстов вида (5.1), составляющих основу ее формирования, специально не удалена информация расщепленных предикатных значений согласно *утверждению 5.3*: этим конструкциям будут соответствовать отдельные области в решетке тезауруса. Для выделения расщеплённых предикатных значений в отдельную решетку с последующим анализом её свойств в этом случае может быть полезным алгоритм сегментации решеток, о котором говорилось в работе [120].

Помимо указанных преимуществ, модель вида (5.2) решает актуальную для нормализации отношений проблему функциональной зависимости неключевых атрибутов от части составного ключа [20, с. 315–319]. Применительно к текстовым базам данных указанная зависимость обусловлена как наличием расщепленных предикатных значений в исходных текстах, так и более широким классом синонимического варьирования в рамках стандартных лексических функций. Оперируя критерием полезности решетки, рассмотренным в главе 4, данную проблему в случае без расщепления лексического значения можно решить либо путем замены слова в тексте на исходное слово-аргумент лексической функции, либо путем выбора того значения ЛФ из нескольких возможных, которое максимизирует полезность решетки.

Следует также отметить еще одну качественную особенность моделей вида (5.2), напрямую связанную с репрезентативностью корпуса текстов, составляющего основу формирования предметных знаний. Как было справедливо отмечено в [183], репрезентативность – это такой тип отображения проблемной области в корпус текстов, при котором последний отражает все свойства проблемной области, релевантные для данного лингвистического исследования. Фактически репрезентативность определяется частотой встречаемости в тексте определенных семантических и синтаксических конструкций из фиксируемых моделью (5.2) и, следовательно, может служить своего рода показателем способности корпуса текстов к сжатию посредством теоретико-решеточного представления.

Связывая репрезентативность исходного корпуса текстов и полезность решетки, отметим, что чем выше репрезентативность корпуса, тем большей полезностью обладает решетка для контекста (5.2), что означает и более высокую степень сжатия по сравнению с линейным представлением текстов. Первостепенную роль здесь играет информативность комбинации слов в составе каждой из рассматриваемых конструкций [183]. Весовой коэффициент информативности здесь вычисляется на основе взаимной зависимости слов в составе конструкции. Хорошим примером может по-

служить поточечный коэффициент взаимной зависимости синтаксически главного w_1 и зависимого слова w_2, обсуждавшийся в [120]:

$$depn(w_2, w_1) = \log_2 \frac{frec(w_2, w_1) \cdot N}{frec(w_2) \cdot frec(w_1)},$$

где $frec(w_2, w_1)$ – частота, с которой w_2 встречается в корпусе как непосредственно синтаксически подчиненное слову w_1; $frec(w_2)$ и $frec(w_1)$ – частоты, с которыми встречаются w_2 и w_1 отдельно в корпусе; N – общее количество слов в корпусе.

Сама репрезентативность корпуса является также показателем отражения в текстах определенного жанра.

Так, для деловой и научной прозы, представленной в формальных решетках на рис. 5.1–5.8, характерно строгое разграничение семантико-синтаксических контекстов вида (4.1) между существительными относительно предикатных слов в составе указанных последовательностей. Пример (из табл. 5.1): *“заниженн-ость завис-ит:от (связан-а:с)”*, но *“уменьшени-е связан-о:с”*. При этом сжатие текстов на основе решётки ФП происходит (в первую очередь) за счет тех предикатных слов, которые либо обозначают ситуации, сходные в той или иной мере по составу участников и характеру выполняемых ими действий, либо (как в приведенном примере) относятся к абстрактной лексике. В целом же способность текстов различных жанров к сжатию является темой отдельного прикладного исследования.

Выводы

Таким образом, в пятой главе предложен метод численной оценки семантической схожести текстов предметного языка относительно ситуаций его употребления. При этом теоретико-решёточное представление ситуации языкового употребления составляет основу кластеризации семантических отношений, формируемых с применением подхода из раздела 3.5.

Использованием теоретико-решёточного представления СЯУ в качестве информационной единицы тезауруса предметной области достигается существенное упрощение процедуры пополнения последнего. Благодаря иерархизации классов СЭ в решётке формальных понятий тезауруса отсутствует необходимость формирования всех без исключения тезаурусных единиц “с нуля”, поскольку часть информации, представляемой текстовыми описаниями фактов действительности уже содержится в тезаурусе на уровне суперпонятий. Сказанное особенно актуально при подготовке тестов открытой формы, когда разработчик теста формулирует один или несколько вариантов “правильного” ответа, опираясь исключительно на собственные знания заданной предметной области.

Отдельной темой для рассмотрения является минимизация тезаурусной единицы разделением языковых и предметных знаний из представляемых ситуацией языкового употребления. Решению этой задачи посвящается заключительная глава работы.

Глава 6
АНАЛИЗ ФОРМАЛЬНЫХ ПОНЯТИЙ И СЖАТИЕ ТЕКСТОВОЙ ИНФОРМАЦИИ РАЗДЕЛЕНИЕМ ПРЕДМЕТНЫХ И ЯЗЫКОВЫХ ЗНАНИЙ

В данной главе показывается, как следует применять методы АФП для оптимального разделения предметных и языковых знаний на основе комплексной методики формирования и кластеризации семантических отношений, изложенной в разделах 3.5, 4.1, 5.2 и 5.3. Вводится понятие смыслового эталона СЯУ и рассматриваются два приближенных метода его построения. Предложен подход к минимизации базы знаний, используемых при анализе семантической схожести фраз предметно-ограниченного ЕЯ, на основе выделения смысловых эталонов. Представлен метод семантической интерпретации ЕЯ-фраз на основе шаблонов, которые строятся для множеств СЭ-фраз по результатам выделения смысловых эталонов СЯУ. Описывается механизм интерпретации ответа обучаемого на тестовое задание открытой формы, а также типовая архитектура системы контроля знаний на основе тестов указанного вида. Основные результаты главы представлены в [64, 95, 96, 174, 175].

6.1. Постановка задачи на примере тестовых заданий открытой формы

В общем случае разработчик теста открытой формы должен описать свой вариант правильного ответа на тестовый вопрос одной либо несколькими ЕЯ-фразами, эквивалентными по смыслу и определяющими ситуацию языкового употребления. При этом не накладывается каких-либо ограничений на исходное множество СЭ-фраз. Тем не менее, при использовании ситуации языкового употребления в качестве информационной единицы тезауруса ЕЯ-фразы, составляющие ее основу, должны максимально точно описывать ситуацию (выражать смысл “на одном дыхании”). Ставится задача разделения знаний о сходных язы-

ковых формах описания различных ситуаций действительности (с одной стороны) и о внешне различающихся формах наиболее “компактного” описания каждой из ситуаций для представления в тезаурусе (с другой стороны).

Для решения данной задачи рассмотрим единицу знаний, представляемую формальным контекстом вида (5.1) и сформированную на основе ЕЯ-фраз, отвечающих вышеуказанному требованию, в качестве смыслового эталона СЯУ. При этом сама модель (5.1) трансформируется к виду:

$$Sk = (Tk, Ks), \tag{6.1}$$

где Ks есть формальный контекст вида (5.1). Множество Tk получается из исходного множества СЭ-фраз заменой каждого слова парой (b_i, f_i), в которой b_i соответствует основе, f_i – флексии этого слова.

Заметим, что среди исходных фраз множества Ts в (1.1) имеются как фразы, определяющие смысловой эталон СЯУ, так и не являющиеся таковыми. Для связи последних с эталоном поставим в соответствие некоторую переменную x_i каждой основе b_i, для которой существует либо признак $m \in Ms : m = p_{bs} \bullet b_i$, либо объект $g \in Gs : g = b_i$. Здесь p_{bs} соответствует символьной константе *“главное – основа:”*, а символом “$\bullet$” обозначается операция конкатенации. При этом на базе двойки (6.1) строится шаблон СЯУ (здесь *“pt”* есть сокр. от англ. *Pattern* – шаблон):

$$Spt = (Idpt, Tpt, Kpt), \tag{6.2}$$

в котором все обозначения основ в составе имен объектов и признаков формального контекста Ks эталона конкретной СЯУ заменяются переменными и задается список конкретизирующих четверок вида

$$(Idpt, Id_S, x_i, b_i), \tag{6.3}$$

где Id_S – идентификационный номер самой СЯУ; $Idpt$ – номер её шаблона. Для СЯУ вновь выделяемого шаблона можно, к примеру, взять $Idpt = Rnd \cdot |Wsx| + 1$, а

$Id_S = \frac{1}{|Wsx|}\left(Rnd \cdot \sum_{i=1}^{|Wsx|} len(b_i) \right)$, где Wsx есть множество основ, конкретизирующих переменные этой СЯУ, len есть длина основы b_i (в символах), $0 \le Rnd < 1$ – случайное число.

Пример формального контекста эталона для СЯУ, определяемой множеством СЭ-фраз на рис. 3.18, показан на рис. 6.1, а соответствующий ему шаблон Kpt формального контекста СЯУ – на рис. 6.2. ЕЯ-фразы исходного множества, вошедшие в эталон, представлены в табл. 3.2. Множество Tpt для рассматриваемого примера показано на рис. 6.3 (каждая пара “основа-флексия” представлена составным объектом *wm* языка Пролог[8]), конкретизации переменных – в табл. 6.1. Следует отметить, что для значительного числа случаев тестирования интерпретация ответа обучаемого состоит в попытке применить шаблон (6.2) “правильного” ответа, сформулированного разработчиком теста. При этом не требуется производить разбор ЕЯ-ответа обучаемого с привлечением внешних программ синтаксического анализа, поскольку достаточно “наложить” анализируемую фразу на один из шаблонов в составе множества Tpt с формированием пар “переменная-основа”, которые сопоставляются с четвёрками вида (6.3) для “правильного” ответа. Сама интерпретация ответа обучаемого происходит за линейное время, не превышающее $|Tpt|$.

[8] Здесь “*wm*” есть сокр. от английского *word marking* – маркировка слова.

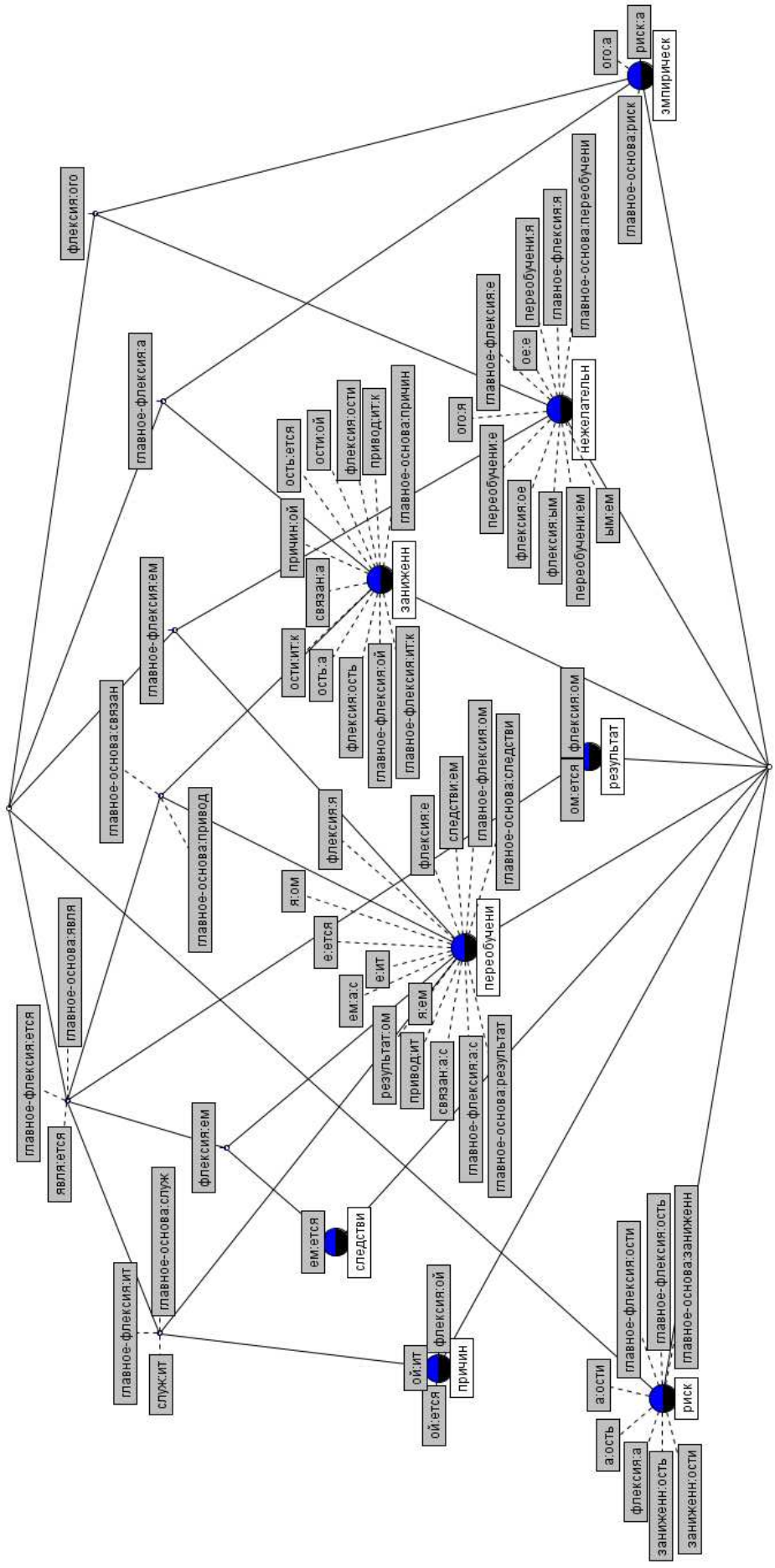


Рис. 6.1. Формальный контекст смыслового эталона для СЭ-фраз на рис. 3.18

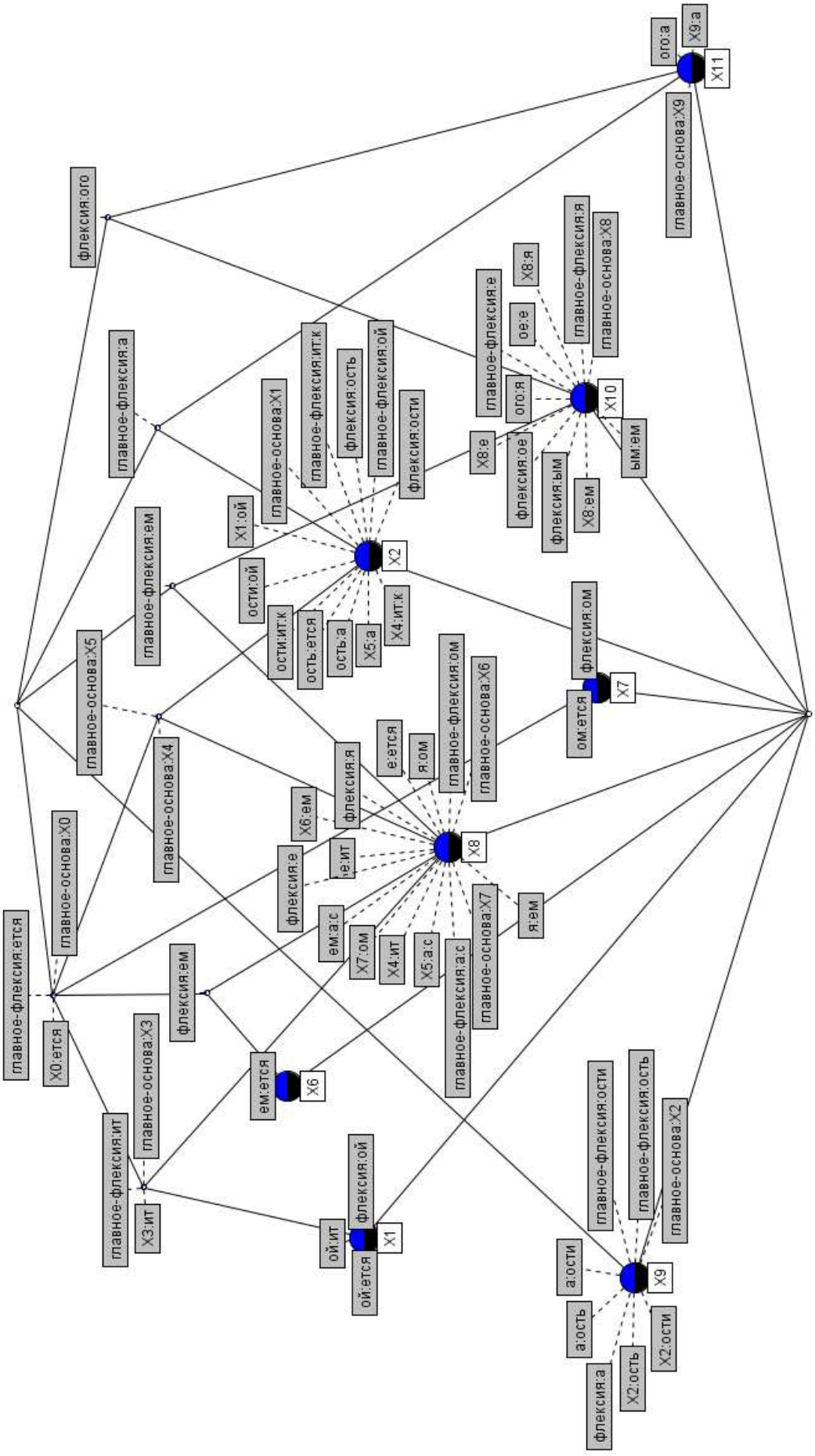


Рис. 6.2. Шаблон формального контекста смыслового эталона СЯУ для СЭ-фраз на рис. 3.18

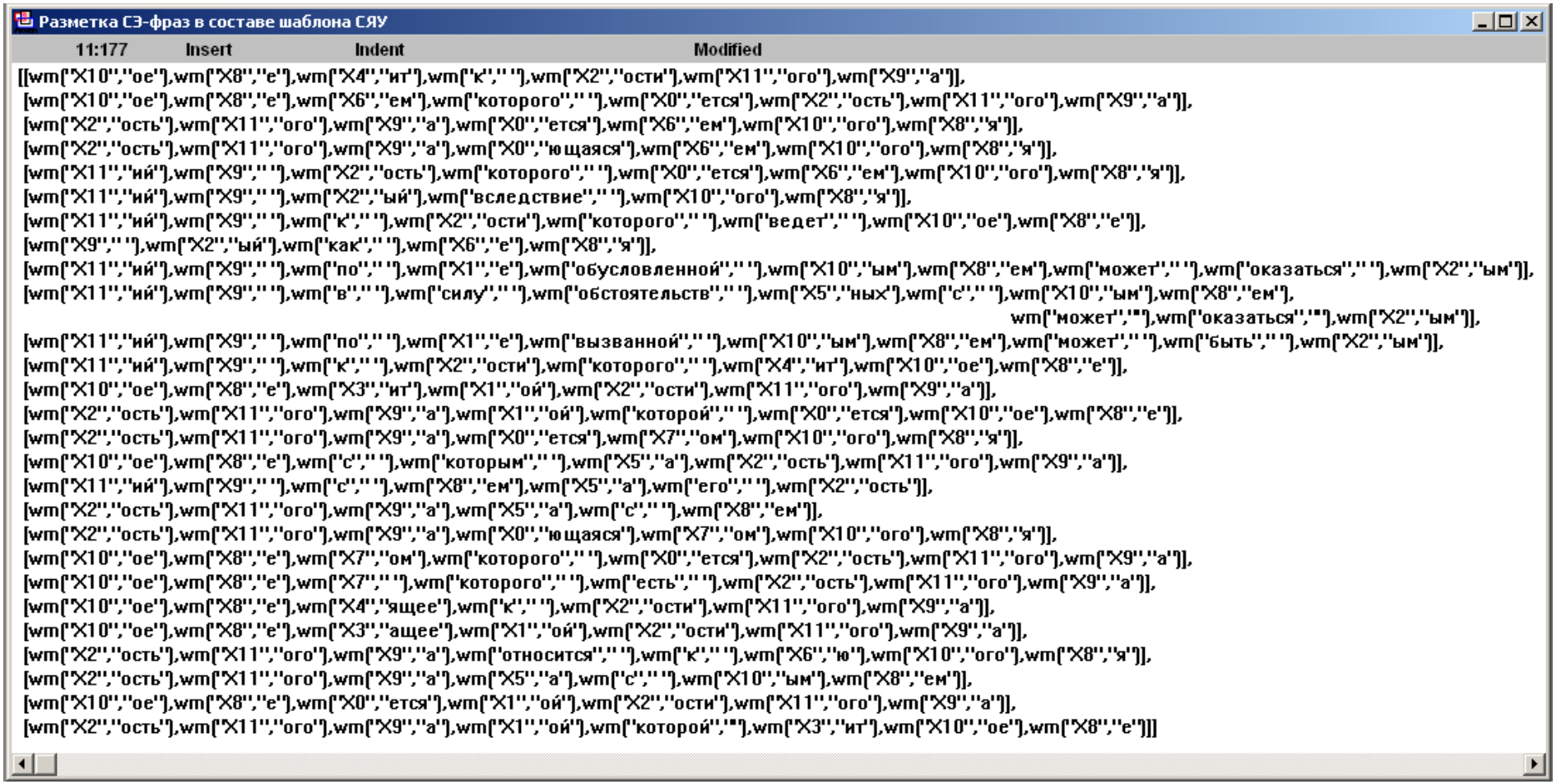


Рис. 6.3. Разметка СЭ-фраз в составе шаблона СЯУ из примера на рис. 3.18

Таблица 6.1

Конкретизация переменных для шаблона СЯУ на рис. 6.2

x_i	b_i	x_i	b_i	x_i	b_i
X0	явля	X4	привод	X8	переобучени
X1	причин	X5	связан	X9	риск
X2	заниженн	X6	следстви	X10	нежелательн
X3	служ	X7	результат	X11	эмпирическ

Следует отметить, что концепция шаблона для СЯУ в виде формального контекста Kpt в составе тройки (6.2) полностью согласуется с определением смысла текста набором характеристических функций, о которых говорилось в разделе 3.1. При этом объекты формального контекста будут составлять области определения и множества значений указанных функций, а классы формальных понятий в решетке для Kpt определяют типы отношений в рамках зависимостей, описываемых λ-выражениями. Сами характеристические функции здесь за счет введения переменных в области определения и множества допустимых значений описывают толкования уже не отдельных лексических значений, а их классов с варьируемой степенью абстракции.

6.2. Формирование смыслового эталона

Шаблоны формальных контекстов в составе троек (6.2) могут быть использованы для синтаксического разбора ЕЯ-фраз из ответа обучаемого. В ходе разбора строится контекст (5.1) относительно некоторой заданной и смежных с ней предметных областей. Конкретизирующие четверки (6.3) по каждой фразе при этом необязательны.

Рассмотрим теперь задачу построения формального контекста самого смыслового эталона как основы моделей (6.1) и (6.2). Как следует из неформального

определения смыслового эталона в начале предыдущего раздела, основой механизма построения указанного контекста может послужить подход к выделению и классификации синтагматических зависимостей, предложенный в разделе 3.5.

Определим условия, при которых объекты и признаки формального контекста $Ke = (Ge, Me, Ie)$, представляющего смысловой эталон, связываются отношением Ie в соответствии с принятыми в указанном разделе обозначениями и соглашениями.

Пусть J есть индексное множество для неизменных частей всех слов, употребленных во всех ЕЯ-фразах из исходного множества СЭ-фраз. Если $\{j,k\} \subset J$ и (j,k) принадлежит множеству ветвей расширенного дерева (3.11), см. *определение 3.5*, то для основ b_j и b_k и флексий f_j и f_k соответствующие им элементы множеств Ge и Me, а также элементы отношения Ie, будут сформированы следующим образом.

Случай 1. Индекс k соответствует родительскому узлу, индекс j – дочернему узлу в расширенном дереве (3.11), а линейная структура ЕЯ-фразы не содержит предлог между словами с индексами j и k.

При этом в состав множества признаков Me формального контекста $Ke = (Ge, Me, Ie)$ будут включены признаки $m_1 = p_{bs} \bullet b_k$, $m_2 = p_{bf} \bullet f_k$, $m_3 = p_{fl} \bullet f_j$ и $m_4 = f_j \bullet ":" \bullet f_k$, основа b_j войдет в множество объектов Ge указанного формального контекста, а пары (b_j, m_1), (b_j, m_2), (b_j, m_3) и (b_j, m_4) войдут в отношение Ie.

Случай 2. Индекс k соответствует родительскому узлу, индекс j – дочернему узлу в расширенном дереве (3.11), линейная структура ЕЯ-фразы содержит предлог p_y между словами с индексами j и k.

В этом случае признаки m_1 и m_3 формируются аналогично *случаю 1*, $m_2 = p_{bf} \bullet f_k \bullet ":" \bullet p_y$, $m_4 = f_j \bullet ":" \bullet f_k \bullet ":" \bullet p_y$, пары (b_j, m_1), (b_j, m_2), (b_j, m_3) и (b_j, m_4) включаются в отношение Ie.

Рекурсивным обходом расширенного дерева вида (3.11) на основе множества СЭ-фраз из примера на рис. 3.18 был, в частности, сформирован формальный контекст, решетка формальных понятий которого представлена на рис. 6.1.

Рассмотрим теперь построение формального контекста вида (5.1) для смыслового эталона по совокупности формальных контекстов указанного вида для отдельных СЭ-фраз, задающих ситуацию языкового употребления. Такая задача актуальна при наличии существенных смысловых ограничений на перифразирование (например, если от обучаемого требуется сохранить авторский язык при пересказе фрагмента художественного произведения в тестах по русской литературе). Положим, что формальные контексты указанной совокупности, упоминаемой далее как список KSE, строятся по результатам синтаксического анализа этих фраз внешней программой, реализующей стратегию разбора на основе наиболее вероятных связей слов. Как и в двух предыдущих главах, возьмем в качестве такой программы “Cognitive Dwarf”. Следует отметить, что данная программа показала высокую точность при решении задачи анализа схожести произведений разных авторов на основе близости частот синтаксических связей и на основе морфологичесих атрибутов слов. Более подробно результаты экспериментов на материале произведений Толстого А.Н., Достоевского Ф.М., Довлатова С.Д., Гоголя Н.В., Гриневского (Грина) А.С., Искандера Ф.А., Толстого Л.Н., Паустовского К.Г., Шукшина В.М., Солженицына А.И. приводятся в [110].

Для решения поставленной нами задачи введем коэффициенты сжатия информации относительно формальных контекстов вида (5.1).

Будем использовать обозначения для подмножеств множества признаков Ms формального контекста (5.1) и их содержательную интерпретацию, принятые нами в разделе 5.2, а именно: M_1 – множество указаний на основу синтаксически

главного слова, M_2 – множество указаний на флексию главного слова, M_3 – множество сочетаний “основа-флексия” для синтаксически главного слова, M_4 – множество сочетаний флексий зависимого и главного слова, M_5 – множество указаний на флексию зависимого слова. Тогда коэффициент сжатия информации по основам для формального контекста указанного вида равен:

$$ks = \frac{\sum_{i=1}^{nbs} ks_i}{nbs}, \tag{6.4}$$

где $ks_i = \dfrac{\sum_{j=1}^{nbs_i}\sum_{k=1}^{nmf} nas_{ijk}}{nbs_i}$; $nbs = |M_1|$; $nmf = |M_2|$;

$$nbs_i = \left|\left\{ g \in Gs : Is(g,m) = true,\ m \in M_1,\ m = p_{bs} \bullet b_i \right\}\right|;$$

$$nas_{ijk} = \left|\left\{ m_k \in M_3 : Is(g_j, m_k) = true,\right.\right.$$

$$\left.\left. \exists\, m_{bf} \in M_2,\ m_{bf} = p_{bf} \bullet f_k,\ m_k = b_i \bullet ":" \bullet f_k \right\}\right|;$$

p_{bf} соответствует символьной константе *“главное – флексия:”*.

Аналогично определяется коэффициент сжатия информации по флексиям:

$$kf = \frac{\sum_{i=1}^{nfs} kf_i}{nfs}, \tag{6.5}$$

где $kf_i = \dfrac{\sum_{j=1}^{nfs_i}\sum_{k=1}^{nmf} naf_{ijk}}{nfs_i}$; $nfs = |M_5|$;

$$nfs_i = \left|\left\{ g \in Gs : Is(g,m) = true,\ m \in M_5,\ m = p_{fl} \bullet f_i \right\}\right|;$$

$$naf_{ijk} = \left|\left\{ m \in M_4 : Is(g_j, m) = true,\right.\right.$$

$$\left.\left. \exists\, m_{bf} \in M_2, m_{bf} = p_{bf} \bullet f_k, m = f_i \bullet ":" \bullet f_k \right\}\right|;$$

p_{fl} соответствует символьной константе “*флексия:*”.

Пусть смысловые эталоны для предметно-языковых знаний эксперта-составителя тестов фиксируются в тезаурусе, представляемом тройкой (5.2). Положим список KSE отсортированным в порядке убывания мощностей множеств объектов для входящих в него формальных контекстов. Тогда построение контекста Ke вида (5.1) для отдельного эталона задается двумя приведенными далее алгоритмами.

Алгоритм 6.1. Выделение потенциальных эталонов.

Вход: KSE ;

Выход: $PE = \{Kpe : Kpe - \textit{ формальный контекст вида}(5.1)\}$;

Начало

Взять очередной $Ks = (Gs, Ms, Is)$ из KSE ;

$Ng_{\max} := |Gs|$;

$PE := \varnothing$;

Начало цикла. Для всех $Ks \in KSE$ *таких, что* $|Gs| = Ng_{\max}$

$KSE_{cur} := KSE \setminus Ks$;

$Kpe := Ks$;

$ks_{\max} := ks(Kpe)$ *согласно формуле* (6.4) ;

$kf_{\max} := kf(Kpe)$ *согласно формуле* (6.5) ;

Начало цикла

Взять очередной $Ks = (Gs, Ms, Is)$ из KSE_{cur} ;

$Kpe_{cur} := Kpe \cup Ks$;

$ks_{cur} := ks(Kpe_{cur})$;

$kf_{cur} := kf(Kpe_{cur})$;

$Flag := ((ks_{cur} > ks_{\max}) \wedge (kf_{cur} > kf_{\max}))$;

При $Flag = false$ *выход из цикла*;

$ks_{\max} := ks_{cur}$;

$kf_{\max} := kf_{cur}$;

$Kpe := Kpe_{cur}$;

Конец цикла;

$PE := PE \cup \{Kpe\}$;

Конец цикла {Для всех $Ks \in KSE$ *таких, что* $|Gs| = Ng_{\max}$ *}*;

Конец {Алгоритм 6.1}.

Замечание. Применительно к паре произвольных формальных контекстов $Kx = (Gx, Mx, Ix)$ и $Ky = (Gy, My, Iy)$ операция объединения $Kx \cup Ky$, известная из теории множеств, интерпретируется как построение формального контекста $Ku = (Gx \cup Gy, Mx \cup My, Ix \cup Iy)$.

Для описания следующего алгоритма необходимо ввести ряд дополнительных обозначений и соглашений. Пусть $CheckAndDel$ есть функция удаления из состава множества объектов каждого формального контекста в списке PE тех объектов, которые встречаются не во всех формальных контекстах данного списка. Те признаки, которые при этом становятся не принадлежащими ни одному объекту, удаляются из множества признаков отдельного формального контекста функцией, обозначаемой далее $Pack$.

Признак будет включен в множество признаков формального контекста эталона, если он входит в состав пятерки признаков $\{m_1, m_2, m_3, m_4, m_5\}$, в которой

$$m_1 = p_{bs} \bullet b,\ m_2 = p_{bf} \bullet f_1,\ m_3 = b \bullet ":" \bullet f_1,\ m_4 = p_{fl} \bullet f_2,\ m_5 = f_2 \bullet ":" \bullet f_1.$$

Здесь b – основа некоторого слова и в соответствии с введенными ранее обозначениями p_{bs} соответствует символьной константе *“главное – основа:”*, p_{bf} – символьной константе *“главное – флексия:”*, p_{fl} – символьной константе

“флексия:”. При этом основе b не должен соответствовать объект формального контекста, если есть другой объект этого же формального контекста, который обладает одновременно признаком m_1 и некоторым другим признаком $m = p_{bs} \bullet b_1$, где $b_1 \neq b$, а основе b_1 не соответствует ни одного объекта этого формального контекста при том, что признак m относится более чем к одному объекту. Функции, которая удаляет из признакового набора каждого объекта формального контекста признаки, не отвечающие данному условию, дадим имя $Closure$. Содержательно данная функция удаляет признаки главных слов-причастий в составе оборотов. Кроме того, указанная функция проверяет принадлежность каждого признака формируемого формального контекста эталона множеству признаков, которые задают последовательности соподчиненных слов по следующему принципу:

$$\begin{cases} \exists\, m_1 \in Me_1 : \left((m_1 = p_{bs} \bullet b) \wedge Ie(g, m_1) \right) = true \\ \exists\, m_2 \in Me_2 : \left(\left(m_2 = p_{bf} \bullet f \right) \wedge Ie(g, m_2) \right) = true \\ \exists\, m_3 \in Me_3 : \left((m_3 = b \bullet ":" \bullet f) \wedge Ie(g, m_3) \right) = true \\ \exists\, m_5 \in Me_5 : \left(\left(m_5 = p_{fl} \bullet f \right) \wedge Ie(b, m_5) \right) = true \end{cases}$$

при максимально возможной длине каждой из последовательностей.

Замечание. Последовательности из трех и более соподчиненных слов, встречающиеся в пятидесяти и более процентах исходных СЭ-фраз из определяющих заданную СЯУ выделяются предварительно на этапе синтаксического разбора и не представлены объектами и признаками формальных контекстов из списка KSE на входе *алгоритма 6.1*. Для каждой такой последовательности по результатам ее синтаксического разбора строится свой формальный контекст (5.1), который будет объединен с формальным контекстом эталона (множество таких формальных контекстов обозначим далее как PSQ). Данный шаг предпринят в целях нежелательного занижения коэффициентов (6.4) и (6.5) при выполнении указанного алгоритма.

Будем использовать символ $Null$ для обозначения формального контекста с пустыми множествами объектов и признаков. Тогда окончательный алгоритм формирования смыслового эталона будет выглядеть следующим образом.

Алгоритм 6.2. Формирование смыслового эталона.

Вход: PE, PSQ;

Выход: Ke – формальный контекст вида (5.1) для эталона;

Начало

$PE_1 := CheckAndDel(PE)$;

$PE_2 := \{ Kpe_2 : Kpe_2 = Pack(Kpe_1), Kpe_1 \in PE_1 \}$;

$Ke_{tmp} := Null$;

Начало цикла. Пока $PE_2 \neq \varnothing$

Взять очередной Kpe_2 из PE_2;

$Ke_{tmp} := Ke_{tmp} \cup Kpe_2$;

$PE_2 := PE_2 \setminus \{ Kpe_2 \}$;

Конец цикла {Пока $PE_2 \neq \varnothing$ *}*;

$Ke := Closure(Ke_{tmp})$;

Начало цикла. Пока $PSQ \neq \varnothing$

Взять очередной Ksq из PSQ;

$Ke := Ke \cup Ksq$;

$PSQ := PSQ \setminus \{ Ksq \}$;

Конец цикла {Пока $PSQ \neq \varnothing$ *}*;

Конец {Алгоритм 6.2}.

Формируемый *алгоритмом 6.2* смысловой эталон соответствует подмножеству максимально проективных (в соответствии с *определением 3.4*) ЕЯ-фраз

исходного СЭ-множества, представляющих лучшие способы описания заданного факта действительности. При этом модель (5.2) позволяет при вычислении функции *Closure* дополнять формируемый эталон информацией слов-синонимов по сходству лексической и флективной сочетаемости на основе ранее сформированных эталонов, уже представленных в тезаурусе.

В качестве иллюстрации далее приводятся результаты совместной работы *алгоритмов 6.1* и *6.2* для СЭ-множества из примера на рис. 3.18, затронутого нами в предыдущем и начале текущего раздела. В целях краткости и большей наглядного изложения исходные формальные контексты приведены не для всех СЭ-фраз, а только для тех из них, которые составили основу построения формального контекста эталона. Заметим, что именно эти фразы приведены в табл. 3.2. как наиболее полно представляющие языковой контекст, фиксируемый тройкой вида (1.1) для рассматриваемой СЯУ. Результирующий формальный контекст *Ke* на выходе *алгоритма 6.2* для всех двадцати семи СЭ-фраз рассматриваемого примера ЕЯ-описания негативных последствий переобучения при скользящем контроле представлен далее на рис. 6.10.

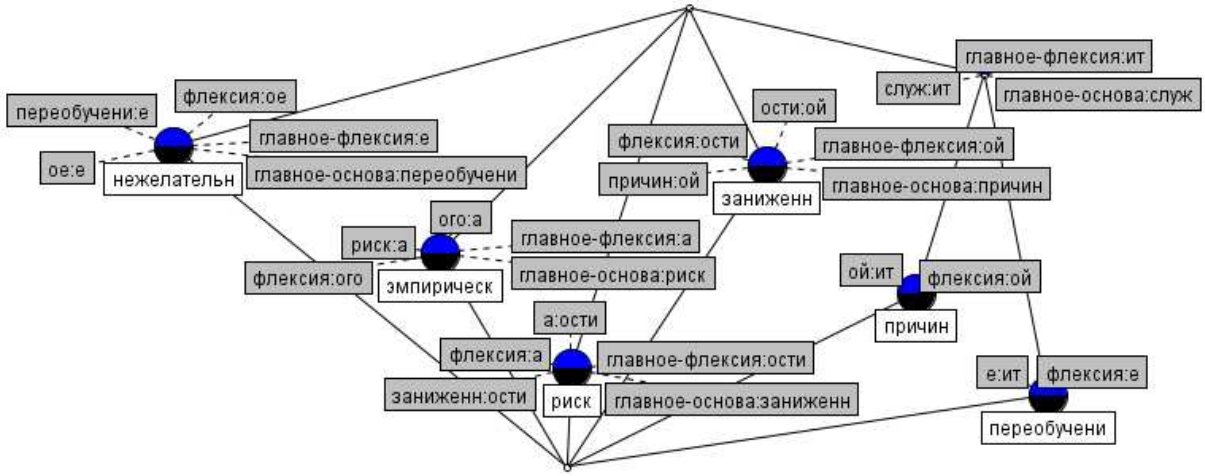


Рис. 6.4. Формальный контекст вида (5.1) для ЕЯ-фразы
“Нежелательное переобучение служит причиной заниженности эмпирического риска”

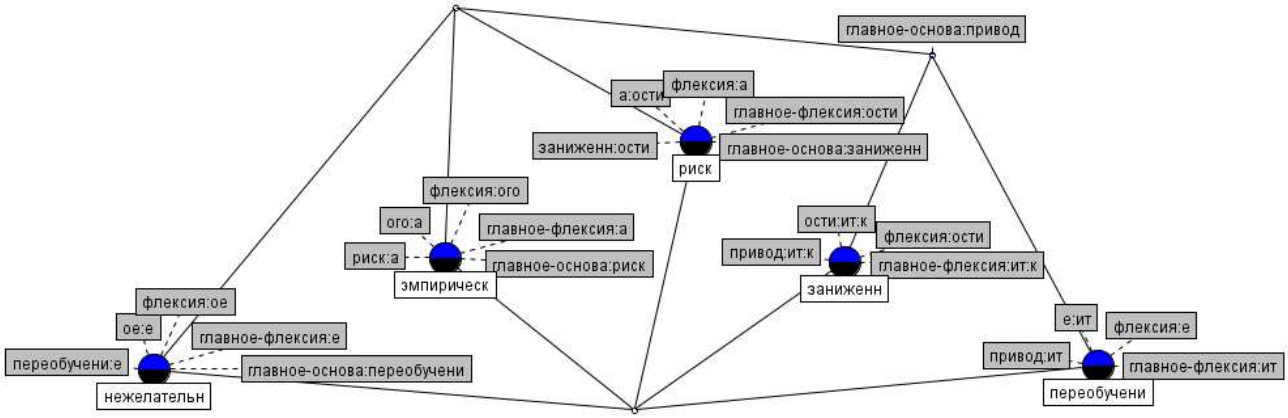


Рис. 6.5. Формальный контекст вида (5.1) для ЕЯ-фразы
“Нежелательное переобучение приводит к заниженности эмпирического риска”

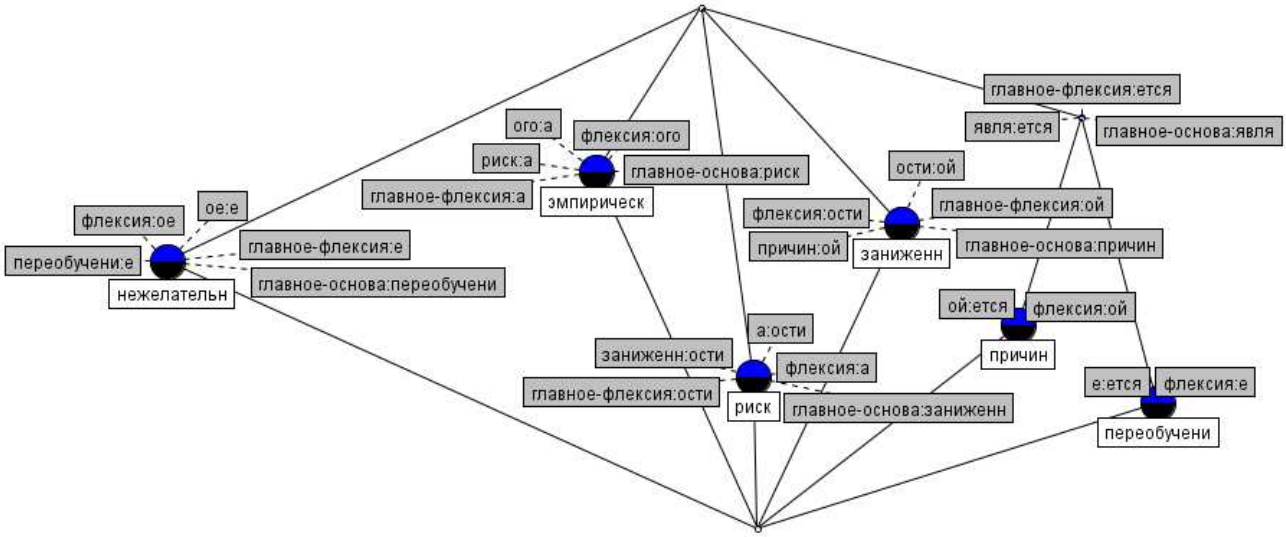


Рис. 6.6. Формальный контекст вида (5.1) для ЕЯ-фразы
“Нежелательное переобучение является причиной заниженности эмпирического риска”

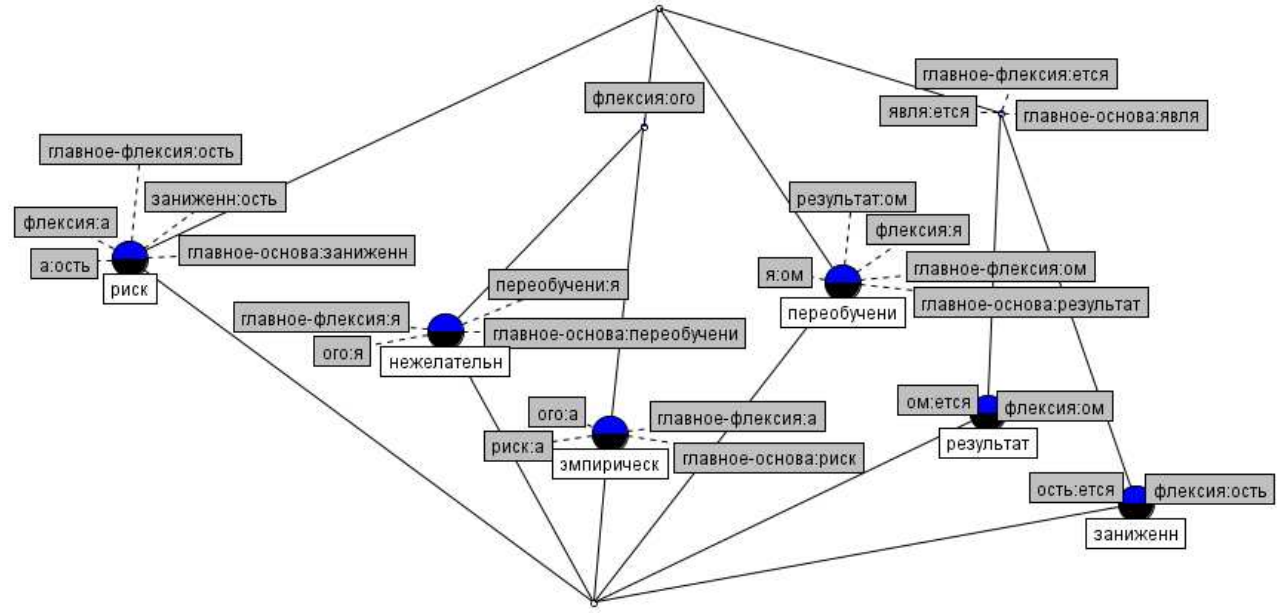


Рис. 6.7. Формальный контекст вида (5.1) для ЕЯ-фразы
“Заниженность эмпирического риска является результатом нежелательного переобучения”

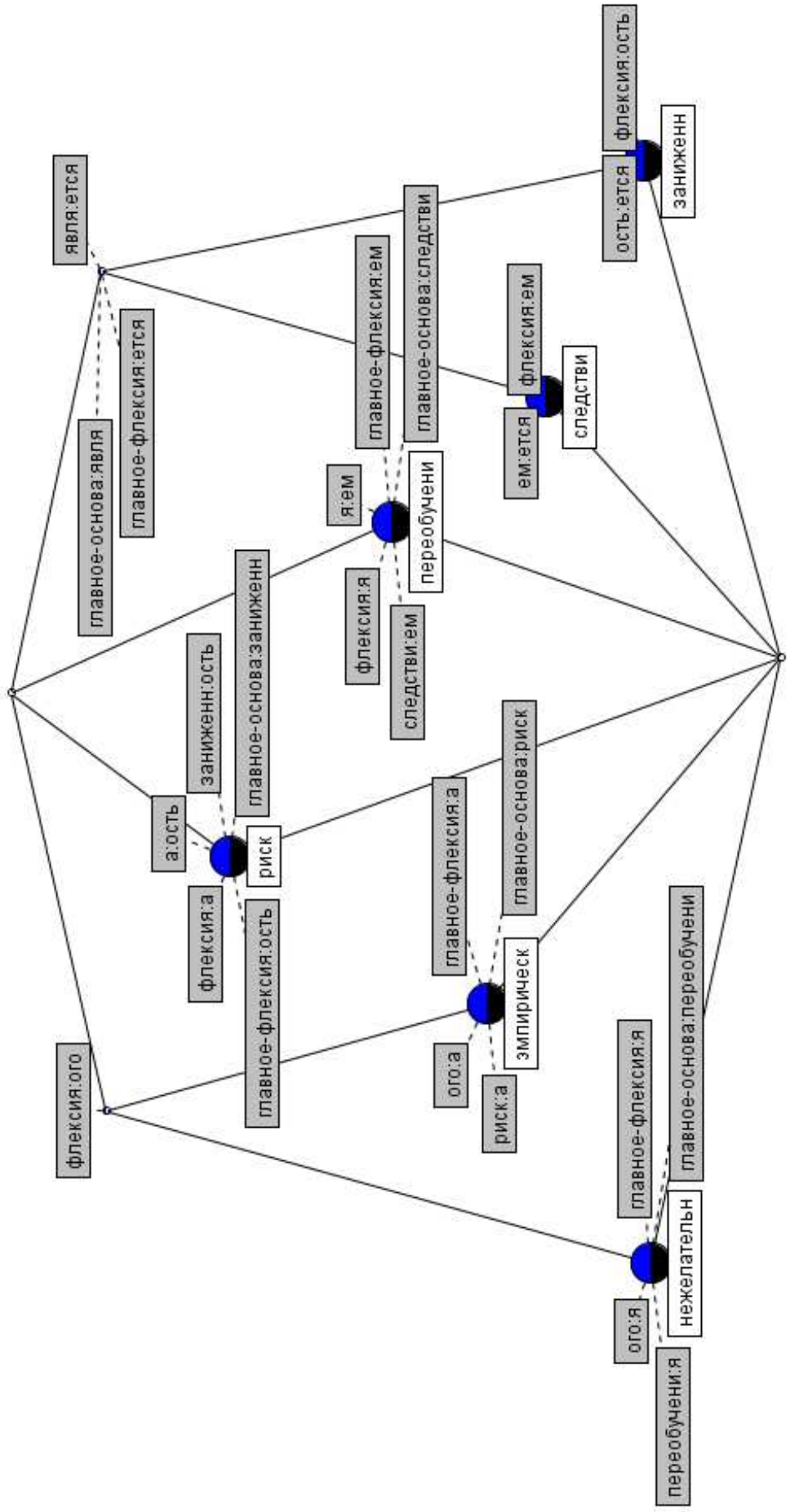


Рис. 6.8. Формальный контекст вида (5.1) для ЕЯ-фразы
“Заниженность эмпирического риска является следствием нежелательного переобучения”

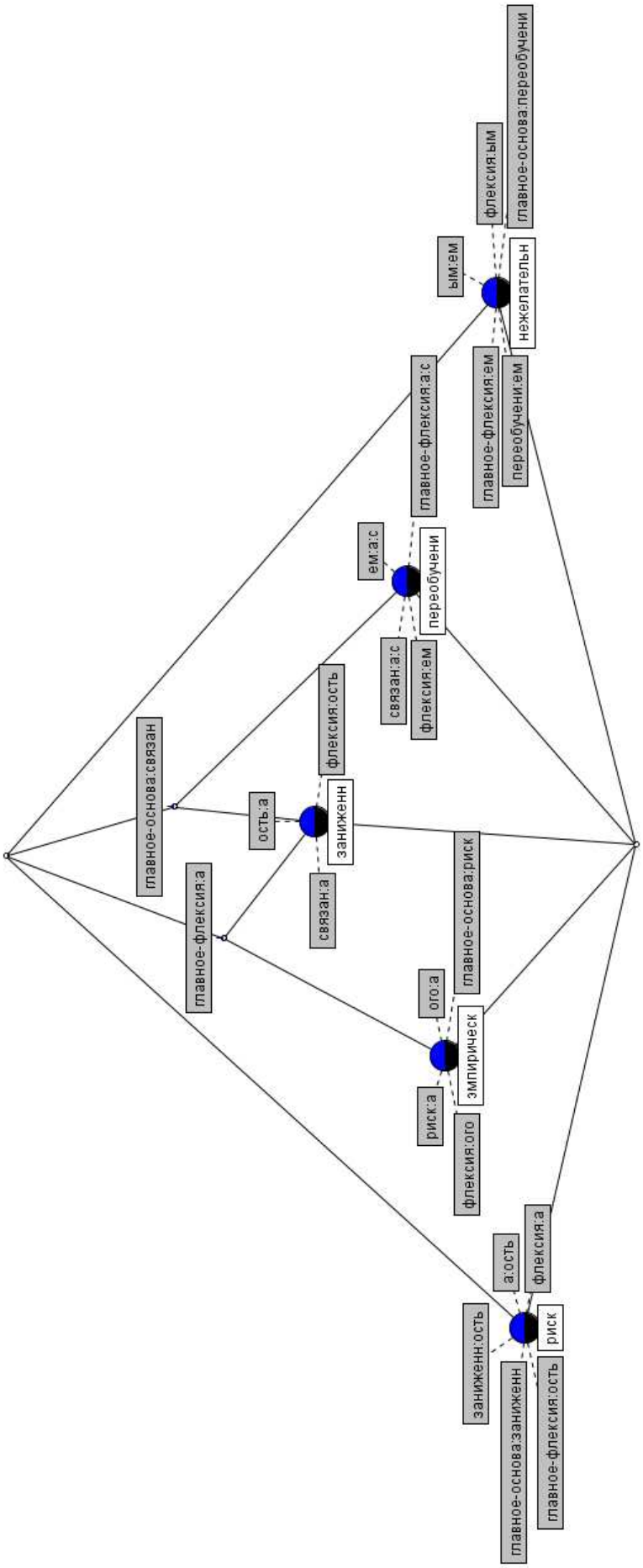


Рис. 6.9. Формальный контекст вида (5.1) для ЕЯ-фразы
“Заниженность эмпирического риска связана с нежелательным переобучением”

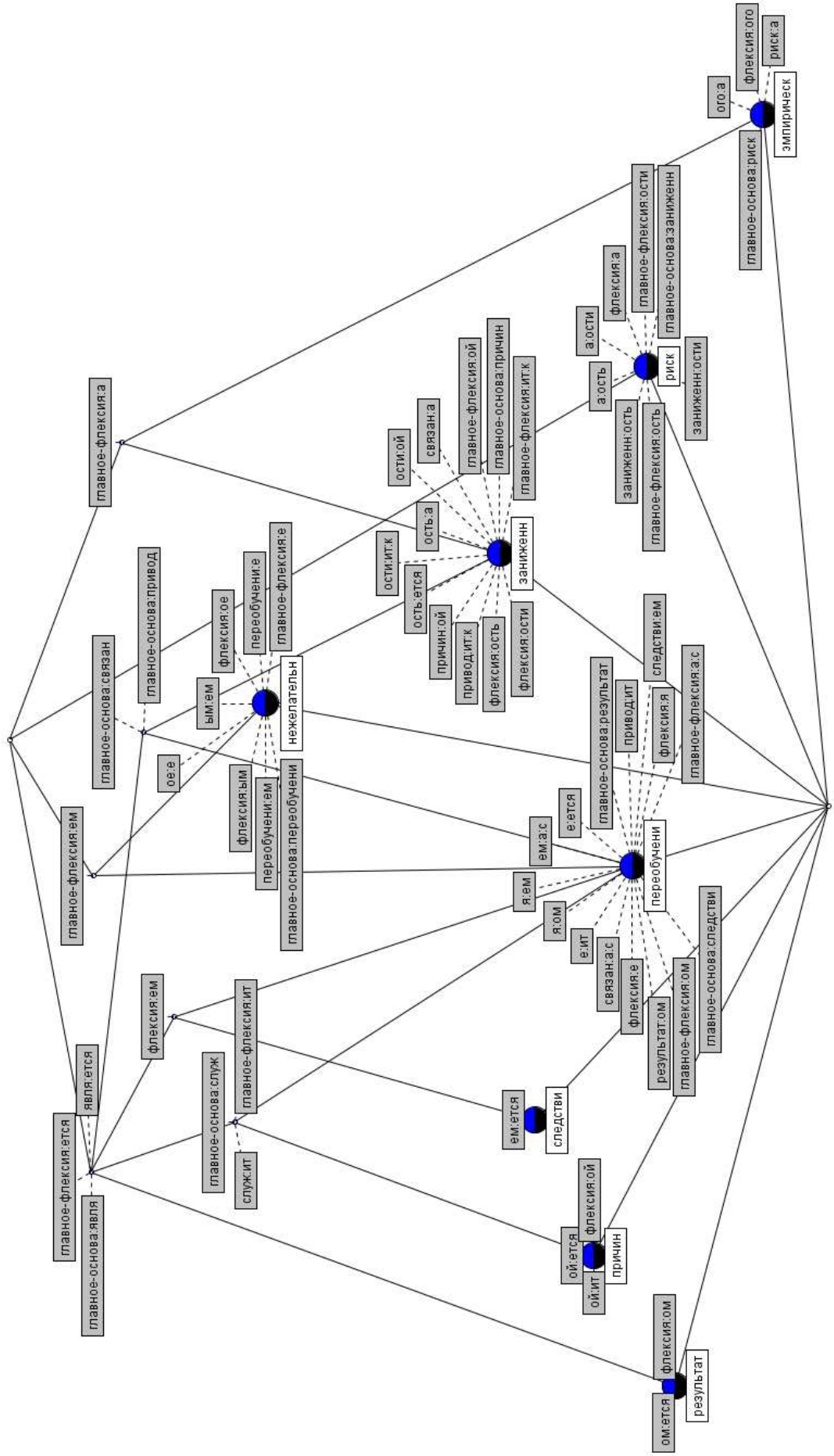


Рис. 6.10. Результирующий формальный контекст вида (5.1) для смыслового эталона

Точность решения, полученного совместной работой *алгоритмов 6.1* и *6.2*, можно оценить погрешностью ε, рассчитываемой по формуле:

$$\varepsilon = \frac{\sum_{i=1}^{|Me|} \frac{Unr_i}{Ne_i}}{|Me|} \cdot 100\%, \qquad (6.6)$$

где Unr_i – число[9] нераспознанных признаков i-го объекта формального контекста Ke на выходе *алгоритма 6.2*; Ne_i – общее число признаков, описывающих i-й объект в составе формального контекста Ke.

Как следует из определения коэффициентов (6.4) и (6.5), значение показателя (6.6) будет тем выше, чем меньше частота, с которой сочетания слов в основе отношения “объект-признак” для формального контекста эталона совместно встречаются в различных СЭ-фразах из задающих рассматриваемую СЯУ. Сказанное полностью соответствует гипотезе о скрытых связях, согласно которой пары слов, встречающиеся в похожих моделях, стремятся иметь близкую семантическую зависимость [186].

Качественной характеристикой процесса формирования смысловых эталонов в целом может послужить соотношение размеров тезауруса, задаваемого моделью (5.2), при построении его на основе формальных контекстов вида (5.1) для всех СЭ-фраз каждой СЯУ и на основе смысловых эталонов при заданном числе ситуаций языкового употребления, представленных в тезаурусе на текущий момент.

В качестве примера на рис. 6.11 указанное соотношение приведено для ситуаций языкового употребления из табл. 6.2, соответствующих ЕЯ-описаниям фактов предметной области “Математические методы обучения по прецедентам”.

[9] Здесь *“unr”* есть сокр. от англ. unrecognized – нераспознанный.

Таблица 6.2

Ситуации языкового употребления

i	Что описывает СЯУ
1	Связь переобучения с эмпирическим риском
2	Связь переусложнения модели с заниженностью средней ошибки на тренировочной выборке
3	Влияние переподгонки на частоту ошибок дерева принятия решений
4	Причина заниженности оценки обобщающей способности алгоритма
5	Зависимость оценки ошибки распознавания от выбора решающего правила
6	Зависимость обобщающей способности логического алгоритма классификации от числа закономерностей алгоритмической композиции

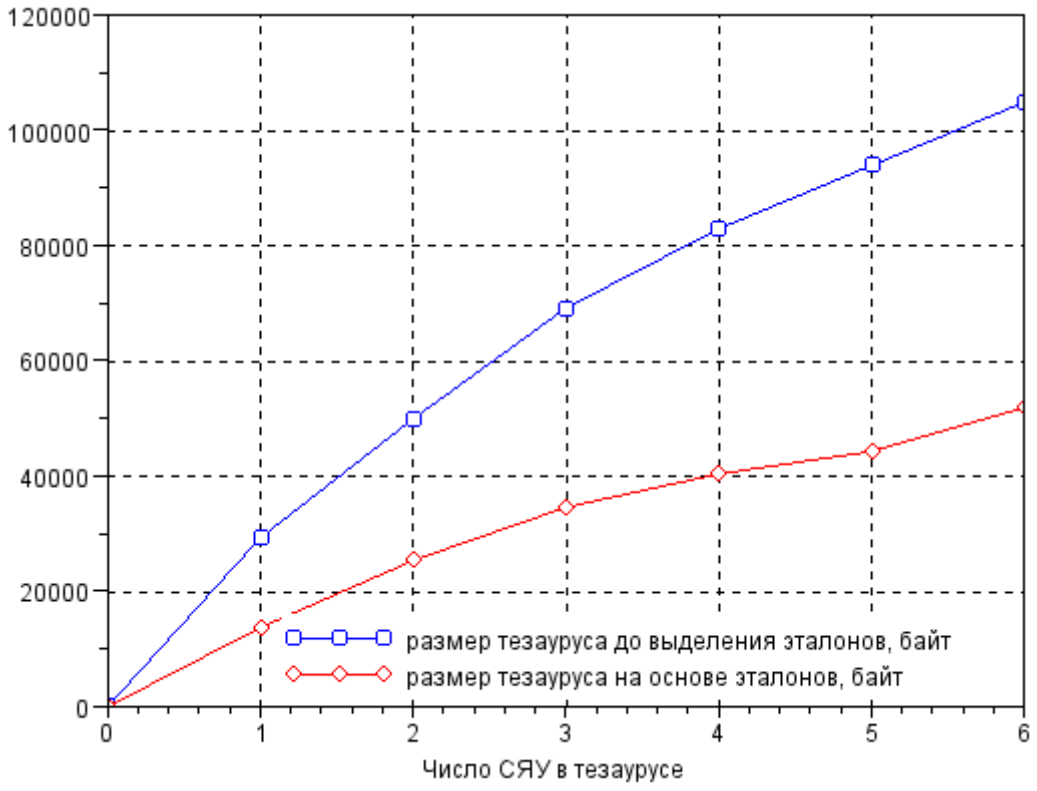


Рис. 6.11. Размер тезауруса для разного числа СЯУ

При этом использование СЯУ в качестве единицы формализованного описания семантики ЕЯ позволяет точно оценить и существенно уменьшить резервируемый объём памяти ЭВМ для хранения текстов с учётом возможных видов синонимии. На сегодняшний день за такую оценку для отдельной фразы из n слов берётся значение $vol(n)=n!$. Введение смысловых эталонов ситуаций языкового употребления позволяет оценивать данный объём сверху как $vol_1(n)=l_1 \cdot n$ и снизу как $vol_2(n)=l_2 \cdot n$, где l_1 – число СЭ-фраз из определяющих СЯУ, из которых l_2 определяют эталон. Соотношение указанных оценок для СЯУ из табл. 6.2 представлено в табл. 6.4.

Таблица 6.3

Фразы максимальной длины из определяющих СЯУ в табл. 6.2

i	Фраза максимальной длины
1	Нежелательное переобучение является причиной заниженности средней величины ошибки алгоритма на обучающей выборке.
2	Тренировочная выборка, на ней проявляется эффект заниженных значений средней ошибки, причиной же является переусложненная модель.
3	Контрольная выборка, принятие деревом решения на ней будет с большей вероятностью ошибки именно по причине переподгонки.
4	Оценка частоты ошибок на выборке, взятой в качестве контрольной, может для алгоритма оказаться заниженной по причине переподгонки.
5	Заниженность оценки ошибки распознавания зависит от выбора правила принятия решений.
6	Число закономерностей алгоритмической композиции влияет на частоту ошибок логического классификационного алгоритма на контрольной выборке.

Таблица 6.4

Оценка объёма памяти для хранения ЕЯ-фразы

i	1	2	3	4	5	6
n	12	15	16	17	10	14
$vol(n)$	$4.790 \cdot 10^8$	$1.308 \cdot 10^{12}$	$2.092 \cdot 10^{13}$	$3.557 \cdot 10^{14}$	$3.629 \cdot 10^6$	$8.718 \cdot 10^{10}$
$vol_1(n)$	648	795	416	442	20	42
$vol_2(n)$	168	225	80	187	20	42

Следует отметить, что часть ситуаций языкового употребления, представленных таблицей 6.2, была задействована нами при построении тезауруса в разделе 5.3 (табл. 5.1). При этом число СЭ-фраз, задающих отдельную СЯУ, в обоих примерах выбиралось экспериментально с целью максимального приближения к реальной ситуации разработки теста открытой формы. Данный показатель представлен в табл. 6.5 параметром N_1, его значение варьировалось в пределах от 2 до 54 для разных фактов рассматриваемой предметной области, i – порядковый номер СЯУ. Для сравнения в этой же таблице приведены значения числа фраз, представляющих эталон (N_2), исходного числа объектов (N_3) и признаков СЯУ (N_4), числа объектов (N_5) и признаков эталона (N_6).

Таблица 6.5

Смысловые эталоны

i	1	2	3	4	5	6
N_1	56	28	29	30	6	10
N_2	8	9	7	9	1	2
N_3	18	17	15	13	12	14
N_4	177	186	173	162	94	81
N_5	9	12	12	11	8	12
N_6	82	90	80	69	35	53

Другой характеристикой процесса формирования эталона является динамика изменения значений коэффициентов сжатия информации в тезаурусе, представляемом формальным контекстом (5.2). По аналогии с коэффициентом (6.4) для формального контекста вида (5.1) коэффициент сжатия информации по основам относительно модели (5.2) определяется как

$$ksth = \frac{\sum_{i=1}^{nbsth} ksth_i}{nbsth}, \tag{6.7}$$

где согласно обозначениям, принятым в разделе 5.3, в частности, для подмножеств множества признаков формального контекста (5.1), $nbsth = |Mth_1|$;

$$ksth_i = \frac{\sum_{k=1}^{nmfth} \sum_{j=1}^{ndm_{ki}} nfms_{ijk}}{nbsth_i};\ nmfth = |Mth_2|;$$

$$nfms_{ijk} = \Big| \{ mth_i \in Mth_3 : Ith(gth_j, mth_i) = true,\quad gth_j \in Gth$$

$$\exists\, m_{bf} \in Mth_2 : m_{bf} = p_{bf} \bullet f_i,\ mth_i = b_i \bullet ":" \bullet f_i,\ Ith(gth_j, m_{bf}) = true,$$

$$\exists\, m_{bs} \in Mth_1 : m_{bs} = p_{bs} \bullet b_i,\ Ith(gth_j, m_{bs}) = true,$$

$$\exists\, mth_k \in M_6 : mth_k = p_b \bullet b_k,\ Ith(gth_j, mth_k) = true,$$

$$\exists\, mth \in M_8 : mth = b_k \bullet b_i,\ Ith(gth_j, mth) = true \} \Big|;$$

$$ndm_{ki} = \Big| \{ gth_j \in Gth : Ith(gth_j, mth) = true,\ mth \in M_8,\ mth = b_k \bullet b_i \} \Big|;\ nmfth = |Mth_2|;$$

$$nbsth_i = \Big| \{ gth \in Gth : Ith(gth, mth) = true,\ mth \in Mth_1,\ mth = p_{bs} \bullet b_i \} \Big|;$$

p_{bf}, p_{bs} и p_b соответствуют символьным константам *“главное – флексия:”*, *“главное – основа:”* и *“основа:”*, соответственно.

По аналогии с коэффициентом (6.5) коэффициент сжатия информации по флексиям относительно формального контекста (5.2) равен

$$kfth = \frac{\sum_{i=1}^{nfsth} kfth_i}{nfsth}, \tag{6.8}$$

где согласно ранее принятым обозначениям $nfsth = |Mth_5|$;

$$kfth_i = \frac{\sum_{j=1}^{nfsth_i} \sum_{k=1}^{nmfth} nafth_{ijk}}{nfsth_i};$$

$$nfsth_i = \left| \{ gth \in Gth : Ith(gth, mth) = true,\ mth \in Mth_5,\ mth = p_{fl} \bullet f_i \} \right|;$$

$$nafth_{ijk} = \left| \{ mth \in Mth_4 : Ith(gth_j, mth) = true, \right.$$

$$\left. \exists\, m_{bf} \in Mth_2 : m_{bf} = p_{bf} \bullet f_k,\ mth = f_i \bullet ":" f_k \} \right|;$$

p_{fl} есть обозначение символьной константы *“флексия:”*.

Графики на рис. 6.12 иллюстрируют динамику изменения значений оценок (6.7) и (6.8) при последовательном добавлении в тезаурус СЯУ из табл. 6.2.

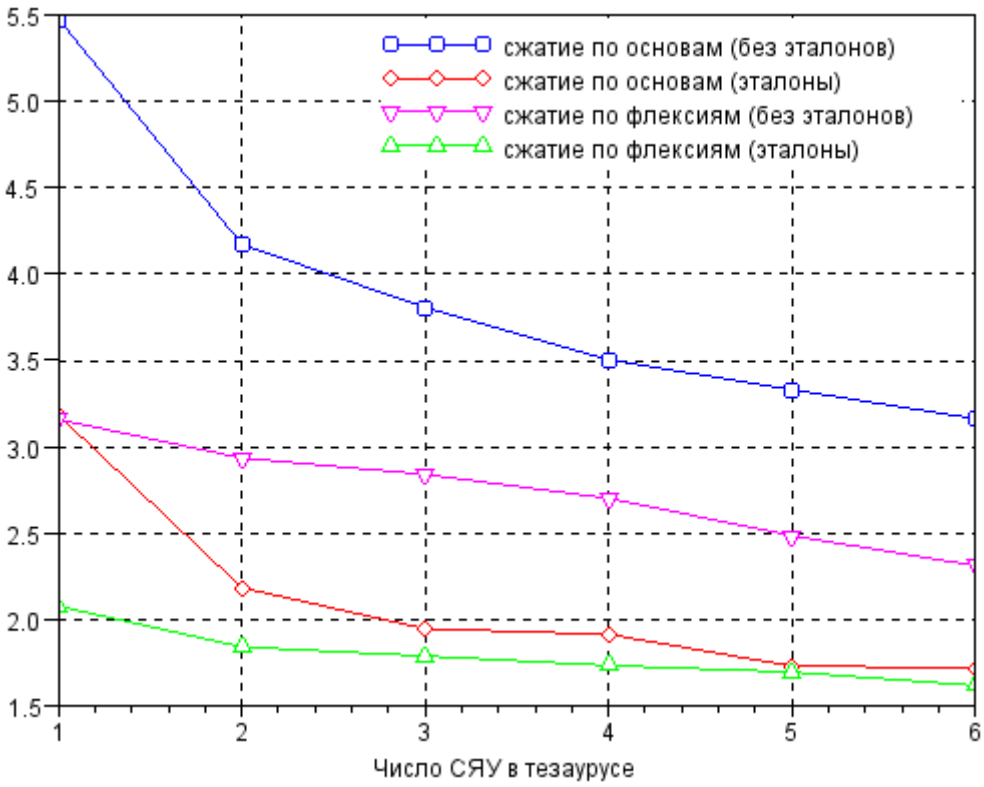


Рис. 6.12. Сжатие информации относительно формального контекста тезауруса

Заметим, что при $Kth = Null$ (тезаурус пуст) для корректной работы функции *CheckAndDel* в составе *алгоритма 6.2* исходные СЭ-фразы первой из определяемых СЯУ ещё не содержат синонимов на уровне словесных обозначений участников (понятий-фигурантов) описываемой ситуации действительности. Здесь задается несколько вариантов рассматриваемой СЯУ, к примеру, один со словом “переобучение”, другой – со словом “переподгонка” (связь переобучения с эмпирическим риском, $i = 1$ в табл. 6.2), формируются смысловые эталоны по

отдельности для каждого варианта, после чего получившиеся формальные контексты объединяются.

В целях повышения точности объектно-признакового описания смыслового эталона, формируемого выделением и классификацией синтагматических зависимостей на основе подхода из раздела 3.5, введём процедуру согласования знаний относительно разных СЯУ, представляемых формальными контекстами (5.1). Само согласование знаний определяется следующим правилом.

Правило 6.1. Пусть b_j – основа слова w, f_j – его флексия, выделенные относительно СЯУ S_j. Предположим, что $w = b_1 \bullet f_1$ для СЯУ S_1, $w = b_2 \bullet f_2$ для СЯУ S_2, причём $b_1 = b_2 \bullet suf$, где suf содержит минимум один символ. Тогда относительно S_1 основа b_1 будет заменена на b_2, флексия f_1 – на $f_3 = suf \bullet f_2$, но только в том случае, если частоты встречаемости флексий f_3 и f_2 в отношениях, представляемых формальным контекстом (5.2) тезауруса заданной предметной области, не уменьшаются при выполнении указанных замен.

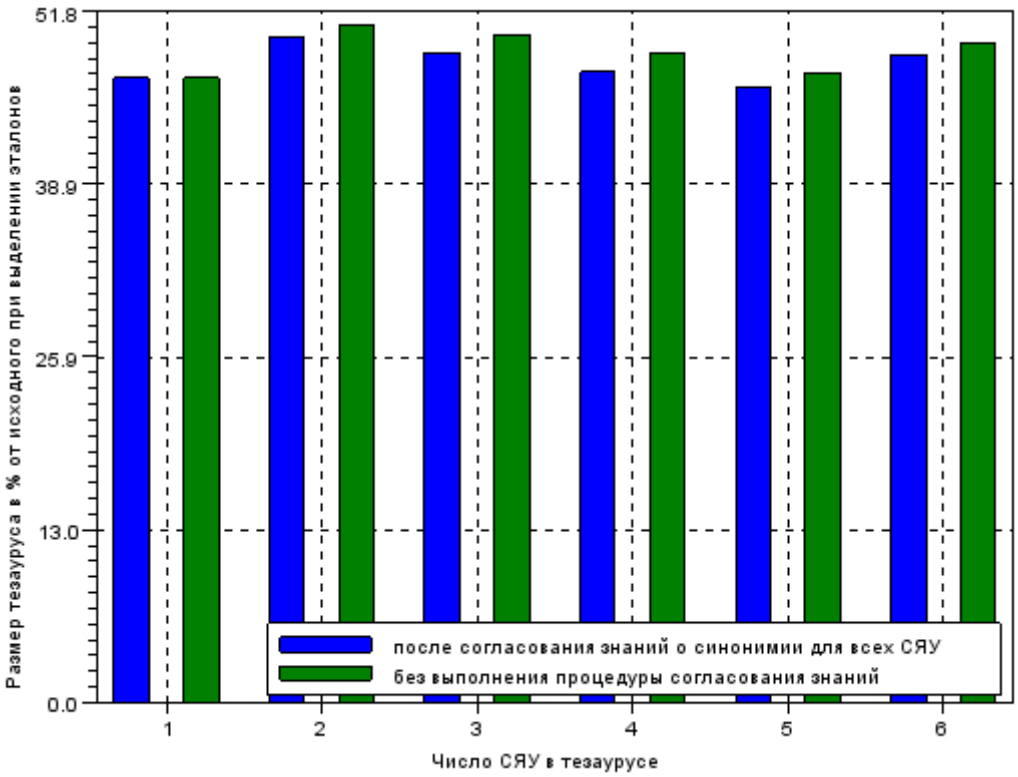


Рис. 6.13. Сокращение размеров тезауруса согласованием знаний по разным СЯУ

Диаграмма на рис. 6.13 иллюстрирует дополнительное сокращение размеров тезауруса в среднем на 1,5% при выполнении указанной процедуры для ситуаций языкового употребления из табл. 6.3. Показателем роста специфичности[10] формальных понятий в решётке тезауруса здесь служит постепенное уменьшение значений коэффициентов (6.7) и (6.8), представленное на рис. 6.14.

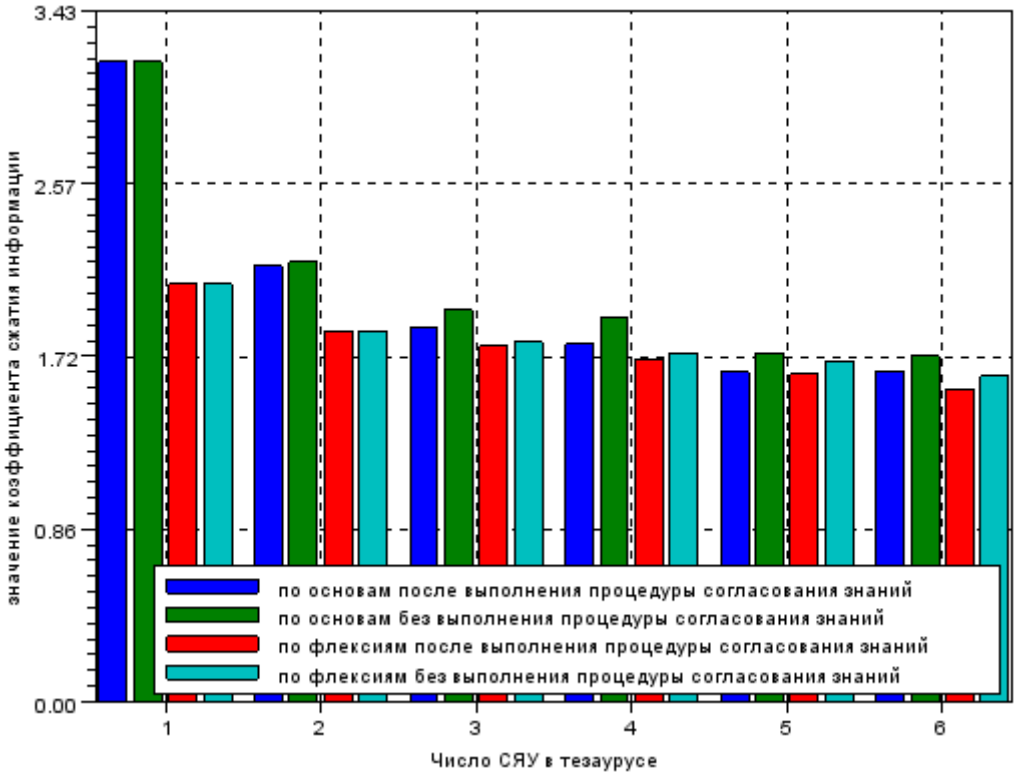


Рис. 6.14. Сжатие информации тезауруса (эталоны выделены)

6.3. Шаблон ситуации языкового употребления и интерпретация текста предметно-ориентированного подмножества естественного языка

Рассмотрим случай отсутствия нужного шаблона вида (6.2) для интерпретации ЕЯ-фразы и анализ возможности найти приближенное решение в виде смыслового эталона путем компиляции формальных контекстов шаблонов нескольких СЯУ.

[10] Под специфичностью формального понятия в данной работе, как и в [116], понимается кратчайшее расстояние до рассматриваемого формального понятия от вершинного формального понятия в решётке.

Пусть имеется множество шаблонов вида (6.2), построенных по результатам выделения эталонов для ситуаций языкового употребления относительно некоторой фиксированной предметной области (содержательно – той предметной области, по которой проводится тестирование знаний). На основе каждого такого шаблона выделяется набор синтаксических отношений, множество всех синтаксических отношений, выделенных по шаблонам СЯУ, далее обозначим как Rp. Отдельное отношение $Rp_i \in Rp$ представляется шестеркой:

$$Rp_i = \left(Idr_i, Tpr_i, Fm_i, Fd_i, Var_i, Mpr_i\right), \tag{6.9}$$

где Idr_i – идентификационный номер отношения Rp_i; Tpr_i – последовательность пар “основа-флексия” для сочетания слов, реализующего отношение Rp_i (в направлении от главного слова к зависимому) в рамках шаблона эталона, при этом $Tpr_i \subset Tpt_j$, $Tpt_j \in Tpt$ в заданном шаблоне вида (6.2); Fm_i и Fd_i – множество возможных вариантов флексии главного и зависимого слова, соответственно, применительно к отношению Rp_i, но уже для всех $Tpt_j \in Tpt$; Var_i – переменная для обозначения основы зависимого слова в составе отношения Rp_i; Mpr_i есть список имен признаков, которые описывают отношение Rp_i в рамках формального контекста Kpt шаблона эталона в соответствующей тройке (6.2). По аналогии с идентификационными номерами для СЯУ и её шаблона $Idr_i = Rnd \cdot \left|Tpt\right|$.

Положим также, что на основе сформированного набора Rp строятся описания возможного присутствия в анализируемой ЕЯ-фразе пар синтаксических отношений, связывающих нераспознанное предикатное слово с непосредственно зявисящими от него словами. Для отдельного нераспознанного слова-предиката такие связи будем представлять четверкой[11]:

[11] Здесь *“Runp”* есть сокр. от англ. Relationship for Unrecognized Predicate word – отношения для нераспознанного предикатного слова.

$$Runp_k = (Idpt, Idr_1, Idr_2, Tunp_k), \qquad (6.10)$$

где $Idpt$ – идентификационный номер шаблона вида (6.2); Idr_1 и Idr_2 – идентификационные номера первого и второго отношения в соответствующих шестёрках (6.9); $Tunp_k : \{(x_1, f_1)\} \bullet Tunp_k \bullet \{(x_2, f_2)\}$ есть последовательность[12] пар “основа-флексия” из некоторой $Tpt_i \in Tpt$ в (6.2), при этом пара (x_1, f_1) соответствует зависимому слову в Tpr_1, а пара (x_2, f_2) – зависимому слову в Tpr_2 и существует $Tpt_j \in Tpt$, $Tpt_i \neq Tpt_j$, такая, что и Tpr_1, и Tpr_2 являются подпоследовательностями[13] в Tpt_j, имея общее главное слово. Пример четвёрки (6.10) и соответствующих ей синтаксических отношений представлены на рис. 6.15, 6.16 и 6.17. Шестёрка (6.9) здесь представлена составным объектом *d_synt_rel*, а четвёрки (6.10) – посредством составного объекта *d_no_marked* языка Пролог[14].

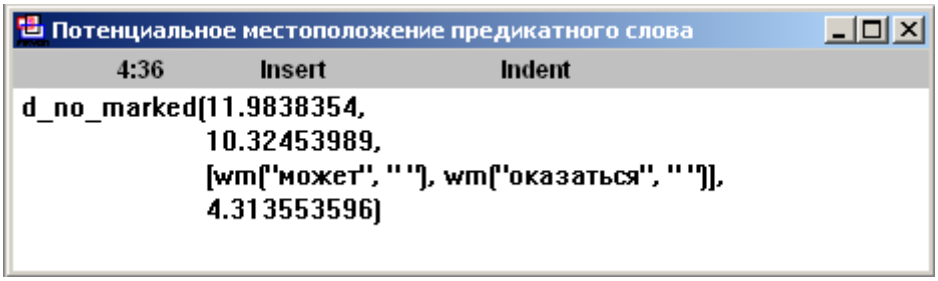


Рис. 6.15. Пример синтаксического окружения для места возможного присутствия предикатного слова

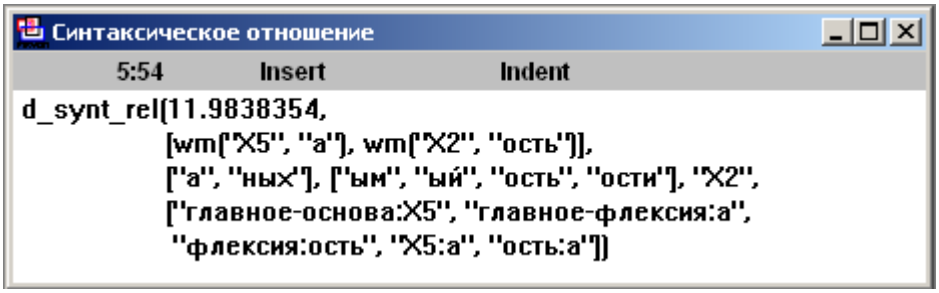


Рис. 6.16. Первая из связей предикатного слова для примера на рис. 6.15

[12] Точкой обозначается рассмотренная нами ранее операция конкатенации символьных строк, здесь она применяется к последовательностям.

[13] с учёирм реверсирования

[14] В примере на рис. 6.15 первый компонент четвёрки (6.10) представлен последним, второй и третий – первым и вторым, стответственно.

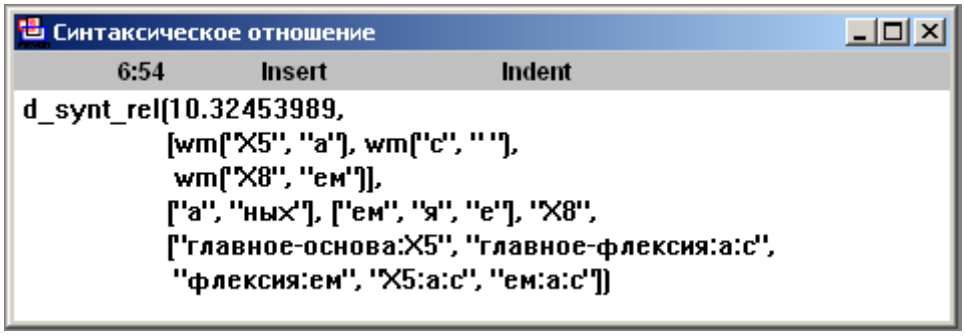


Рис. 6.17. Вторая из связей предикатного слова для примера на рис. 6.15

Правило 6.2. Обозначим множество четверок вида (6.10), выделенных по шаблонам для совокупности СЯУ, как $Runp$, а последовательность слов интерпретируемой ЕЯ-фразы – как WK. Тогда при наличии множеств Rp и $Runp$ относительно заданного множества шаблонов вида (6.2) построение формального контекста (5.1) для смыслового эталона анализируемой ЕЯ-фразы производится рекурсивно выделением в исходной последовательности WK некоторой совокупности слов WK_{mid}, отвечающей одному из следующих условий:

(1) WK_{mid} – подпоследовательность[15] WK, $WK = WK_{bef} \bullet WK_{mid} \bullet WK_{rest}$ (для дальнейших рассуждений обозначим последний элемент последовательности WK_{bef} как w_1, $w_1 = b_1 \bullet f_1$, а первый элемент в WK_{rest} – как w_2, $w_2 = b_2 \bullet f_2$)[16] и $\exists\left(Runp_k \in Runp, Rp_1 \in Rp, Rp_2 \in Rp\right)$, где $Tunp_k = WK_{mid}$, а для заданного $Idpt$ существуют четвёрки вида (6.3), которые ставят основы слов в соответствие переменным в составе пар из Tpr_1 и Tpr_2. Первые элементы последовательностей Tpr_1 и Tpr_2 совпадают и равны $\left(x_p, f_p\right)$, а последние есть $\left(x_1, f_1\right)$ и $\left(x_2, f_2\right)$, соответственно, причем переменная x_1 конкретизируется основой b_1, а переменная x_2 – основой b_2. При этом в формируемый формальный контекст эталона будут добавлены объекты b_1 и b_2,

[15] Здесь возможен вариант подпоследовательности с учетом реверсирования.

[16] Сокращенные обозначения *“bef”* и *“mid”* – от английских *before* и *middle*, соответственно.

множество признаков для добавляемых объектов будут составлять элементы списков Mpr_1 и Mpr_2, соответственно, где переменные заменены их значениями из конкретизирующих четвёрок (6.3) для заданного $Idpt$. Дальнейшее построение формального контекста эталона идёт для последовательностей $WK_{bef} \bullet \{b_p \bullet f_p\}$ и $\{b_p \bullet f_p\} \bullet WK_{rest}$, где b_p есть основа, конкретизирующая переменную для первых элементов последовательностей Tpr_1 и Tpr_2.

(2) $WK_{mid} = \{w_p, w_1\}$, w_p и w_1 не обязательно образуют подпоследовательность в WK (в том числе с учетом реверсирования), но $\exists Rp_i \in Rp$ такое, что имеются четвёрки вида (6.3), которые ставят основы слов в соответствие переменным в составе пар из Tpr_i, при этом в рамках Rp_i слово w_p идентифицируется как главное, слово w_1 – как зависимое, $w_p = b_p \bullet f_p$, $w_1 = b_1 \bullet f_1$, $Tpr_i = \{(x_p, f_p), (x_1, f_1)\}$, переменная x_p конкретизируется основой b_p, а переменная x_1 – основой b_1. В формируемый формальный контекст эталона добавляется объект b_1, множество признаков для добавляемого объекта будут составлять элементы списка Mpr_i, в которых переменные заменены их значениями.

(3) $WK_{mid} = \{w_p, p_y, w_1\}$, требования к w_p и w_1 аналогичны *условию (2)* за исключением того, что в рамках Rp_i w_p связываются с w_1 через предлог p_y. Добавление информации в формальный контекст эталона для данного, а также для последующих трех условий происходит по аналогии с выполняемым по *условию (2)*.

(4) $WK_{mid} = \{w_p, w_1\}$, требования аналогичны *условию (2)* за исключением того, что $Tpr_i = \{(x_p, fp_p), (x_1, fp_1)\}$, $w_p = b_p \bullet f_p$, $w_1 = b_1 \bullet f_1$, а $((fp_p = f_p) \wedge (fp_1 = f_1)) \neq true$, но при этом $f_p \in Fm_i$ и $f_1 \in Fd_i$.

(5) $WK_{mid} = \{w_p, p_y, w_1\}$, p_y – предлог, требования аналогичны *условию (3)* за исключением того, что $Tpr_i = \{(x_p, fp_p), (p_y, ""), (x_1, fp_1)\}$, $w_p = b_p \bullet f_p$, $w_1 = b_1 \bullet f_1$, а $((fp_p = f_p) \wedge (fp_1 = f_1)) \neq true$, но при этом $f_p \in Fm_i$ и $f_1 \in Fd_i$.

(6) $WK_{mid} = \{w_p, w_1\}$, здесь как и в четырех предыдущих условиях, w_p и w_1 не обязательно образуют подпоследовательность в WK (в том числе с учетом реверсирования) и существует $Rp_i \in Rp$ такое, что имеются четвёрки вида (6.3), которые ставят основы слов в соответствие переменным в составе пар из Tpr_i, но относительно фиксированной ситуации языкового употребления. При этом $Tpr_i = \{(x_p, fp_p), (x_1, fp_1)\}$, $w_p = b_p \bullet f_p$, $w_1 = b_1 \bullet f_1$, а $((fp_p = f_p) \wedge (fp_1 = f_1)) \neq true$ и либо $f_p \in Fm_i$, либо $f_1 \in Fd_i$.

В ходе каждого последующего рекурсивного прохода процедуры построения формального контекста эталона при выполнении любого из перечисленных шести условий те слова, для которых уже найдены связи, удаляются из списка ещё не рассмотренных. Перед запуском на выполнение рассматриваемой процедуры в этот список заносятся все слова из WK. Когда указанный список становится пустым, происходит выход из процедуры и выдача сформированного формального контекста в качестве результата. Помимо того, в каждом рекурсивном проходе для *условий (2)–(6)* идет запоминание пары либо тройки слов, относительно которых устанавливается отношение, во избежание зацикливания.

В качестве примера рассмотрим построение смыслового эталона в виде формального контекста (5.1) для простого распространенного предложения *“Нежелательное переобучение служит причиной заниженности средней ошибки на тренировочной выборке”*. Положим, что текущее содержимое базы знаний не позволяет интерпретировать эту фразу посредством одного из шаблонов (6.2), но имеются шестёрки вида (6.9), представленные на рис. 6.18–6.29[17] и соответст-

[17] Как и на рис. 6.16–6.17, второй компонент шестёрки (6.9) представлен спиком составных объектов *wm* языка Пролог, см. также рис. 6.3.

вующие синтаксическим отношениям в рамках ЕЯ-фраз *“Переусложнение модели служит причиной заниженности средней ошибки на тренировочной выборке”* и *“Нежелательное переобучение служит причиной заниженности эмпирического риска”*. Кроме того, для всех переменных в составе рассматриваемых шестёрок (6.9) имеются конкретизации, представленные в табл. 6.6 и соответствующие двум указанным фразам.

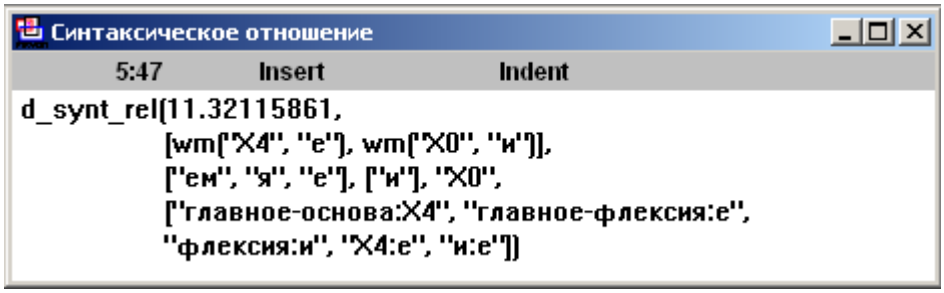


Рис. 6.18. Синтаксическая связь для *“переусложнение модели”*

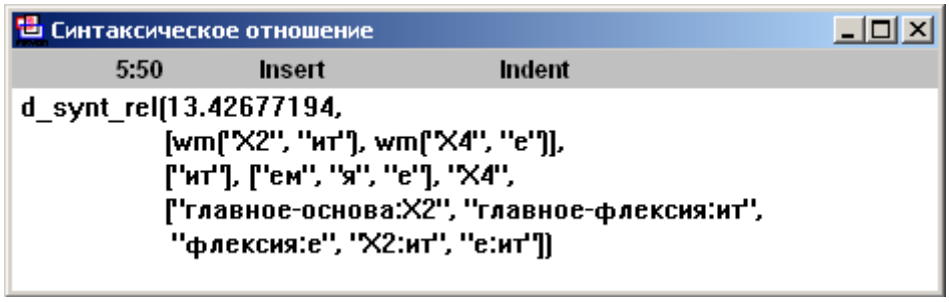


Рис. 6.19. Синтаксическая связь для *“служит переусложнение”*

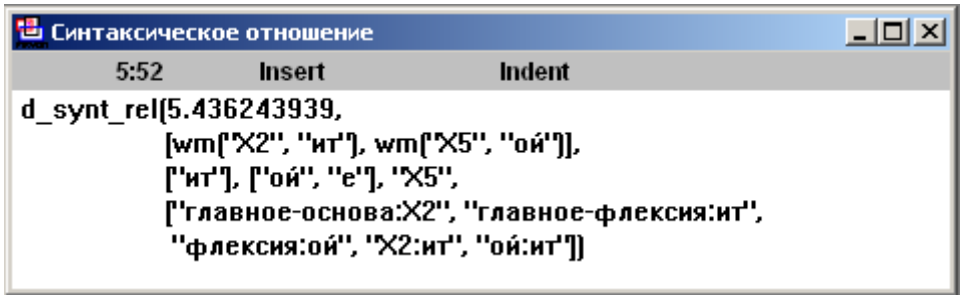


Рис. 6.20. Синтаксическая связь для *“служит причиной”* (вариант 1)

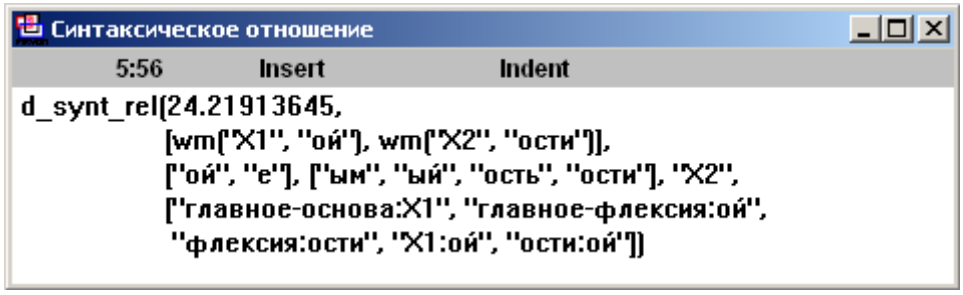


Рис. 6.21. Синтаксическая связь для *"причиной заниженности"*

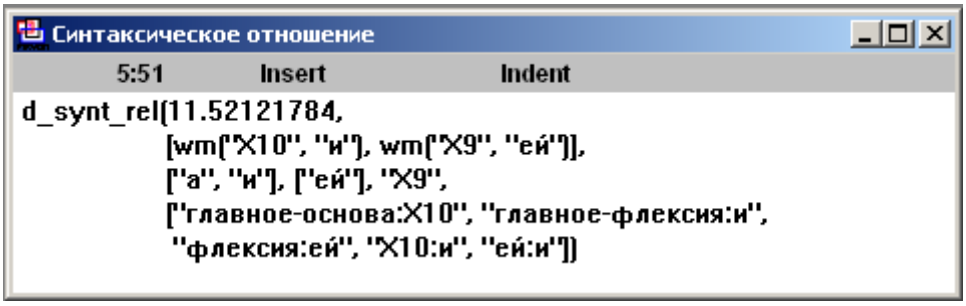


Рис. 6.22. Синтаксическая связь для *"ошибки средней"*

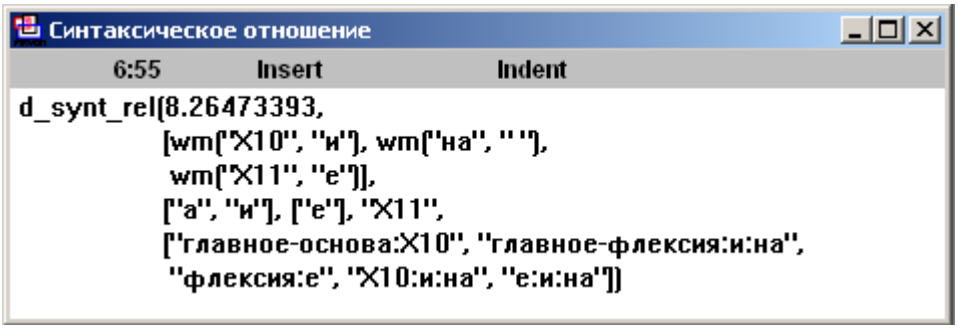


Рис. 6.23. Синтаксическая связь для *"ошибки на выборке"*

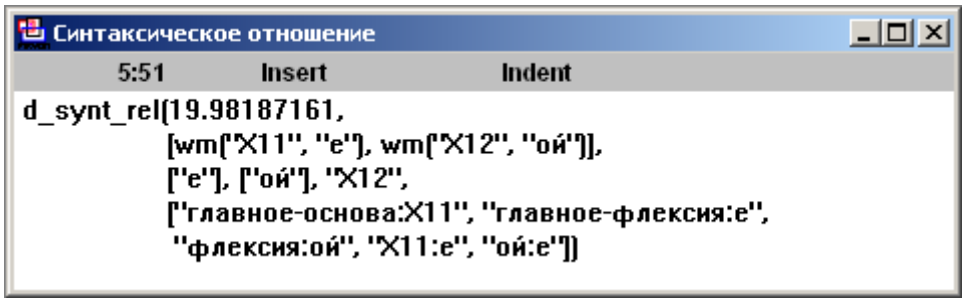


Рис. 6.24. Синтаксическая связь для *"выборке тренировочной"*

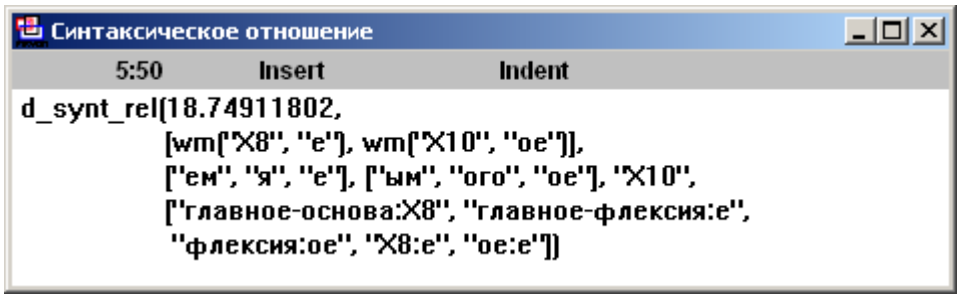


Рис. 6.25. Синтаксическая связь для *“переобучение нежелательное”*

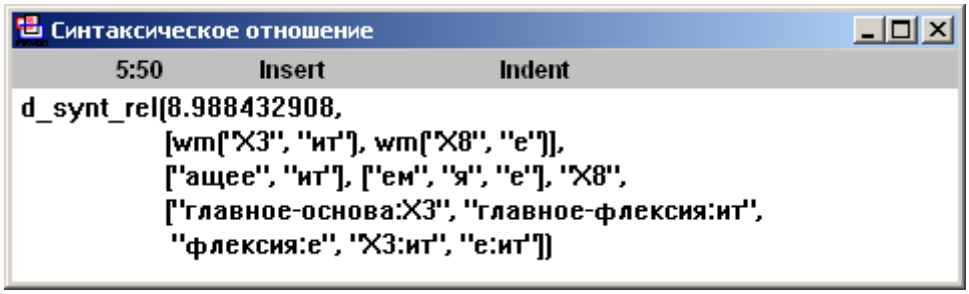


Рис. 6.26. Синтаксическая связь для *“служит переобучение”*

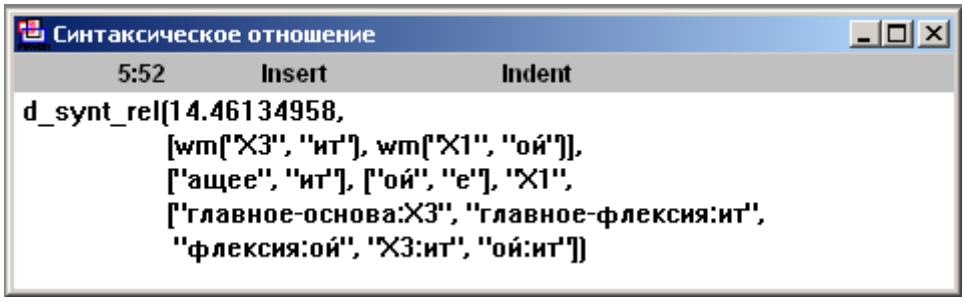


Рис. 6.27. Синтаксическая связь для *“служит причиной”* (вариант 2)

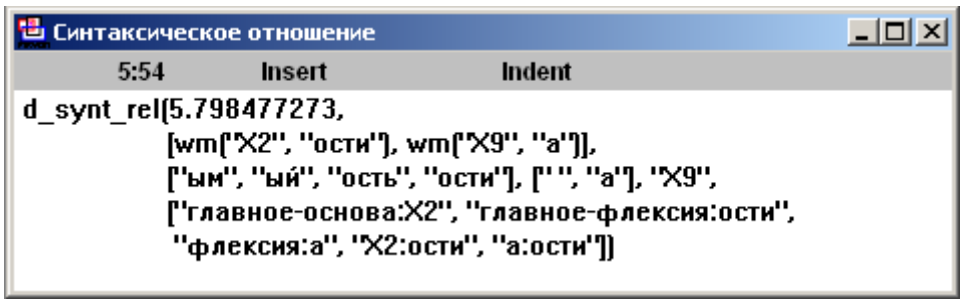


Рис. 6.28. Синтаксическая связь для *“заниженности риска”*

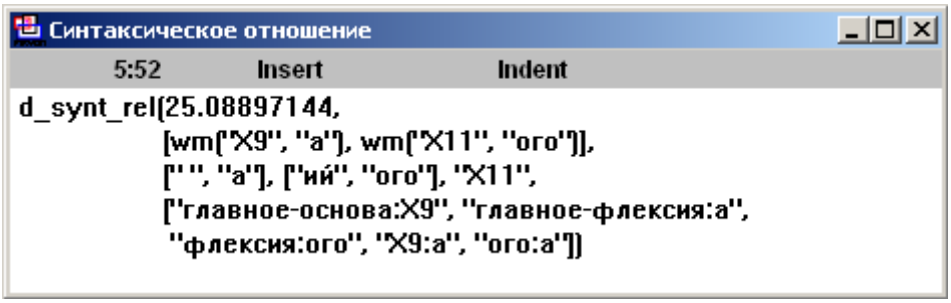


Рис. 6.29. Синтаксическая связь для *“риска эмпирического”*

Таблица 6.6

Конкретизации переменных для примеров на рис. 6.18–6.30

x_i	основа	№ СЯУ по табл. 6.2	x_i	основа	№ СЯУ по табл. 6.2
X0	модел	2	X1	причин	1
X2	служ	2	X2	заниженн	1
X4	переусложнени	2	X3	служ	1
X5	причин	2	X8	переобучени	1
X9	средн	2	X9	риск	1
X11	выборк	2	X10	нежелательн	1
X12	тренировочн	2	X11	эмпирическ	1
X6	заниженн	2	X10	ошибк	2

Пусть для словосочетания *“заниженности ошибки”* в рамках СЯУ, соответствующей ЕЯ-фразе *“Переусложнение модели служит причиной заниженности средней ошибки на тренировочной выборке.”*, в базе знаний не найдена информация, представляемая шестёркой вида (6.9), в которой второй компонент $Tpr_i = \{(x_p, "ости"), (x_1, "и")\}$ при наличии конкретизирующих четверок (6.3) для пар $(x_p, "заниженн")$ и $(x_1, "ошибк")$. В то же время существует синтаксическое отношение, описываемое шестёркой (6.9) и представленное составным Пролог-объектом на рис. 6.30, где переменная *X6* конкретизирована основой *“заниженн”*, а *X10* – основой *“ошибк”* относительно указанной СЯУ, причем флексия *“и”* входит в список возможных вариантов флексии зависимого слова.

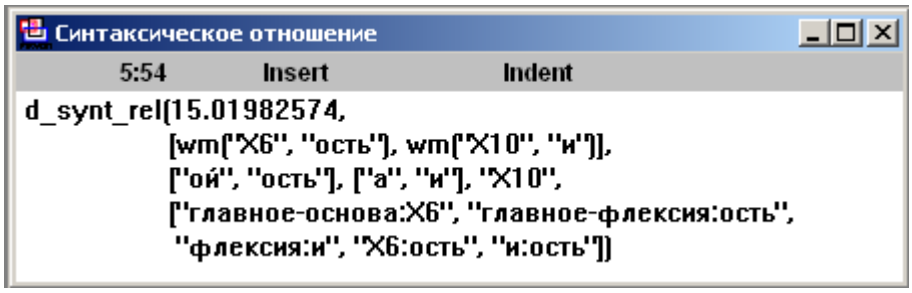


Рис. 6.30. Синтаксическая связь для *“заниженность ошибки”*

Тогда построение искомого формального контекста эталона обеспечивается выполнением условий *правила 6.2* для сочетаний слов в составе анализируемого предложения так, как показано в табл. 6.7. Результирующий формальный контекст представлен решеткой на рис. 6.31.

Таблица 6.7

Соответствие сочетаний слов условиям *правила 6.2*

сочетание слов	№ условия	сочетание слов	№ условия
нежелательное переобучение	2	*заниженности ошибки*	6
переобучение служит	2	*ошибки средней*	2
служит причиной	2	*ошибки на выборке*	3
причиной заниженности	2	*выборке тренировочной*	2

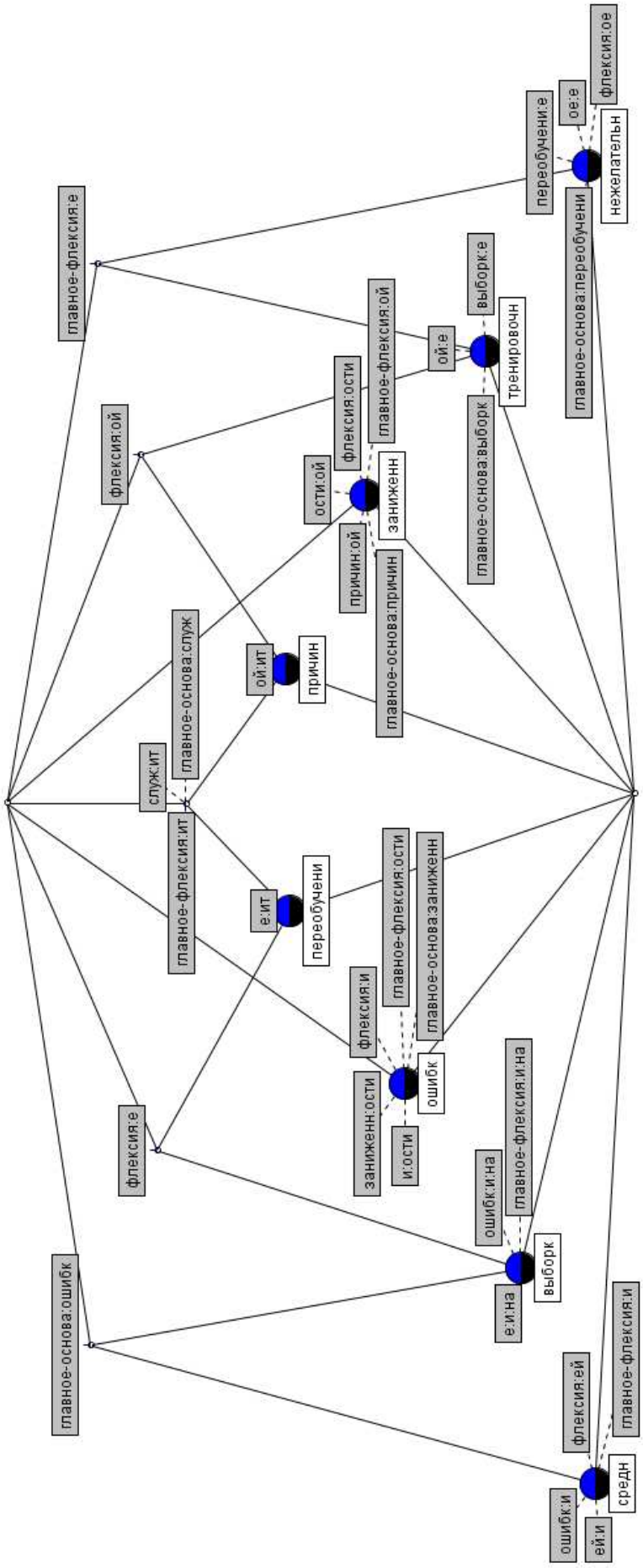


Рис. 6.31. Формальный контекст смыслового эталона по результатам компиляции шаблонов двух СЯУ

Следует отметить, что *правило 6.2* применимо для построения шаблона формального контекста СЯУ анализируемого предложения. В этом случае вместо основ слов в составе имен объектов и признаков формируемого формального контекста будут использованы переменные, причем каждой переменной ставится в соответствие в точности одна основа, а имя каждой переменной должно быть уникальным вне зависимости от существующих имен переменных в описаниях компонент шестёрок (6.9). По каждой переменной, задействуемой в строящемся шаблоне формального контекста, задаётся конкретизирующая четверка (6.3) относительно СЯУ анализируемого предложения на основе конкретизаций вида (6.3), которые используются при построении формального контекста смыслового эталона того же предложения в соответствии с *правилом 6.2*.

6.4. Типовая архитектура системы контроля знаний с применением тестовых заданий открытой формы

Рассмотрим основные требования, которым должна отвечать подиситема обработки ЕЯ автоматизированной системы тестирования знаний на основе заданий открытой формы, и оценим на адекватность этим требованиям предложенный в настоящей главе механизм интерпретации ответа обучаемого.

Во-первых, анализатор текста подсистемы обработки конструкций ЕЯ здесь должен быть приспособлен к обработке “неграмматичностей”, то есть высказываний с отклонениями от грамматической нормы, характерных для диалогов между носителями флективного языка со свободным порядком слов в предложении. За счет описания синтаксических отношений шестёрками вида (6.9), включающих в качестве обязательных компонент множества возможных окончаний для главного и зависимого слова, предложенный механизм интерпретации ответа позволяет устанавливать связь между словами анализируемого ЕЯ-высказывания на основе знаний о лексических и флективных сочетаниях в раз-

личных контекстах. Наличие грамматической правильности предложений при этом не является обязательным.

Во-вторых, интерпретация результатов тестирования требует наличия сложной проблемной области, представляемой базой предметных знаний или заранее сформированным множеством правильных ответов. Формирование таких наборов данных требует оперирования большим по объёму множеством сущностей и, как следствие, наличия единых механизмов обработки информации. Предложенная в настоящей главе концепция смыслового эталона в виде формального контекста отвечает указанному требованию благодаря унифицированному теоретико-решёточному представлению анализируемого ЕЯ-ответа тестируемого и экспертных знаний, фиксируемых тезаурусом.

Учитывая указанные требования и особенности предложенного механизма интерпретации ответа на тестовое заданий открытой формы, в составе системы контроля знаний следует выделить тринадцать основных компонентов, представленных на рис. 6.32. Эти компоненты следующие:

- БФЭ – блок формирования эталонов;
- БФШ – блок формирования шаблонов;
- БСШ – блок слияния шаблонов;
- БФТ – блок формирования тезауруса;
- БВТ – блок выбора теста;
- ТЕСТ – блок выполнения теста;
- БФЗ – блок формирования заданий, которые помещаются в базу данных, именуемую как ЗАДАНИЯ;
- компоненты ТЕЗАУРУС, КОНКРЕТИЗАЦИИ и ШАБЛОНЫ составляют базу предметно-языковых знаний системы. Сюда же входит СО – база Синтаксических Отношений, представляемых шестёрками (6.9) совместно с четвёрками (6.10) и формируемых посредством БФО – блока формирования отношений на основе предварительно сформированной базы шаблонов (6.2).

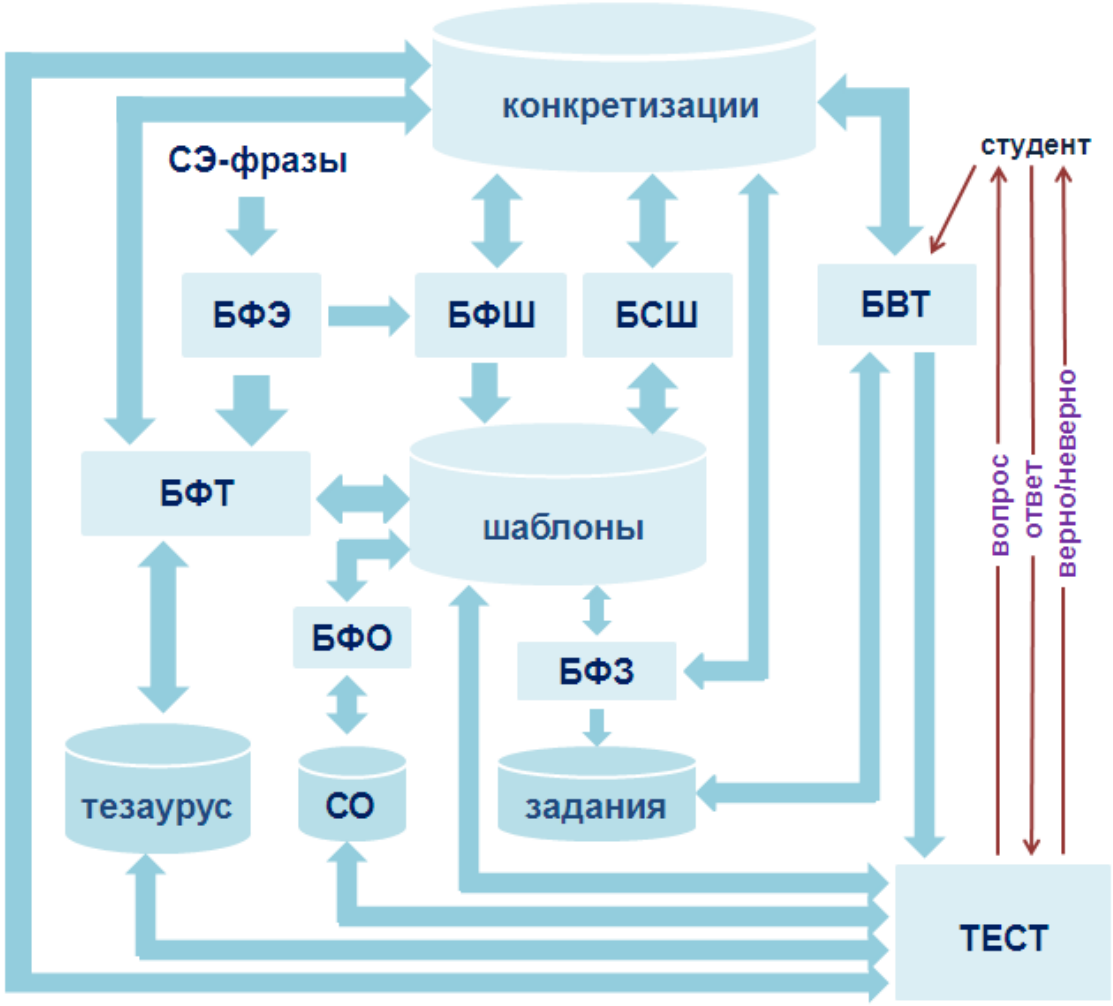


Рис. 6.32. Архитектура системы тестирования знаний

Рассмотрим более детально назначение каждого из представленных на рис. 6.32 компонентов системы тестирования знаний.

Исходными данными для формирования предметно-языковых знаний системы служит совокупность множеств СЭ-фраз, на основе каждого из которых блок формирования эталонов строит свой эталон в виде формального контекста (5.1). Последний поступает в блок формирования шаблонов, где на основе поступившего эталона строится шаблон вида (6.2) с занесением соответствующих четверок (6.3) в базу конкретизаций. Блок формирования тезауруса заносит единицы знаний, представляемые формальными контекстами вида (5.1), в тезаурусную базу согласно требованиям модели (5.2). При этом сам тезаурус вполне может хранить не смысловые эталоны, а их шаблоны. В этом случае занесение кон-

кретизирующих четверок (6.3) по каждой СЯУ в базу конкретизаций является обязательным.

Вне зависимости от формы представления единиц знаний в тезаурусе (смысловые эталоны или их шаблоны) назначение базы шаблонов – хранение шаблонов СЯУ, описываемых посредством троек (6.2). При этом блок слияния шаблонов для пары шаблонов $Spt_1 = (Idpt_1, Tpt_1, Kpt_1)$ и $Spt_2 = (Idpt_2, Tpt_2, Kpt_2)$ указанного вида, представленных в базе шаблонов, строит новый шаблон $Spt_3 = (Idpt_3, Tpt_3, Kpt_3)$: $Tpt_3 = Tpt_1 \cup Tpt_2$, $Kpt_3 = Kpt_1 \cup Kpt_2$ (см. замечание к *алгоритму 6.1*) тогда и только тогда, когда на всех имеющихся в базе конкретизациях для каждой СЯУ, соответствующей Spt_1, найдется СЯУ, соответствующая Spt_2 и описывающая тот же факт действительности (по мнению носителя ЕЯ). Запуск процедуры слияния шаблонов инициируется пользователем.

Назначение блока формирования заданий – организация взаимодействия системы и преподавателя-эксперта по заданной предметной области в процессе составления теста. Каждое тестовое задание представляет собой совокупность вопроса и шаблона (6.2) для правильного ответа на вопрос плюс соответствующие конкретизирующие четверки (6.3) из базы конкретизаций. При этом текст вопроса вводится в специальном текстовом редакторе, также включаемом в состав системы, а введённому вопросу преподаватель тут же ставит в соответствие нужный шаблон из предварительно сформированных вместе с сопутствующими четвёрками вида (6.3). С учетом особенностей реальных тестов задания целесообразно объединять в пакеты, при этом в базе ЗАДАНИЯ на рис. 6.32 будут отдельно храниться сами задания и пакеты заданий для организации тестов с варьируемой степенью сложности. Блок выбора теста должен предоставлять возможность пользователю-тестируемому (студенту) работы как с отдельными заданиями, так и с пакетами, что особенно актуально при подготовке к испытаниям в рамках Единого Государственного Экзамена.

В соответствии с выбранными студентом заданиями блок ТЕСТ реализует, собственно, саму процедуру тестирования. Ответ на вопрос тестового задания

тестируемй вводит в текстовом редакторе, аналогичном используемому блоком формирования заданий. После того, как испытуемый ввёл ответ, система делает попытку применить шаблон (6.2) “правильного” ответа с учетом заданных для этого шаблона конкретизирующих четверок (6.3) в рамках задания. При успешном сопоставлении ответ тестируемого идентифицируется как верный, работа с заданием заканчивается и осуществляется либо переход к следующему заданию в пакете (если число заданий более одного), либо выход. Если сопоставление закончилось неуспешно, делается попытка применить другие шаблоны из базы и в случае успеха – доказать наличие отношения схожести между ситуациями языкового употребления для ответа обучаемого и для “правильного” ответа в соответствии с *определением 5.1*. При успешном доказательстве вычисляется оценка схожести СЯУ по формуле (5.5)[18]. Полученное численное значение используется как основа выставления тестируемым оценок, например, по традиционной пятибалльной шкале, а также для сбора статистики. В случае отсутствия подходящего шаблона в базе делается попытка найти приближённое решение в виде формального контекста (5.1) для ответа тестируемого с применением рекурсивной процедуры на основе *правила 6.2*, описанной в разделе 6.3. Если применение указанной процедуры проходит успешно, то доказывается наличие отношения схожести между ситуациями языкового употребления “правильного” ответа и ответа обучаемого в соответствии с *определением 5.1*. Построенный с применением рекурсивной процедуры формальный контекст (5.1) при этом представляет СЯУ ответа обучаемого. Далее при успешном доказательстве вычисляется оценка схожести СЯУ по формуле (5.5). При неуспешном применении процедуры на основе *правила 6.2*, а также при неуспешном доказательстве схожести СЯУ значение указанной оценки здесь принимается равным нулю, а ответ тестируемого идентифицируется как неверный.

[18] В случае успешного сопоставления шаблону правильного ответа значение схожести равно единице согласно *определению 5.1* и формуле (5.5).

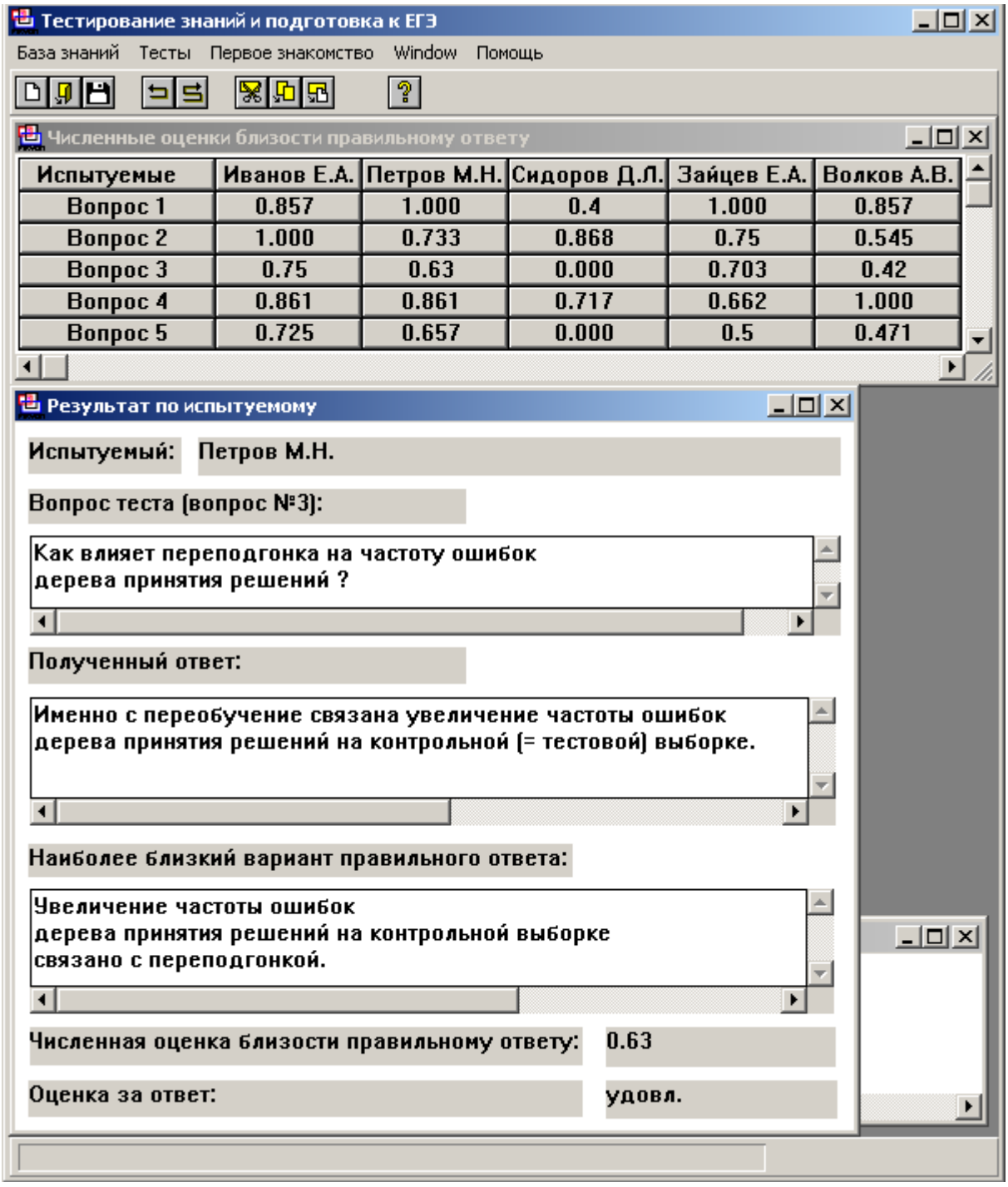
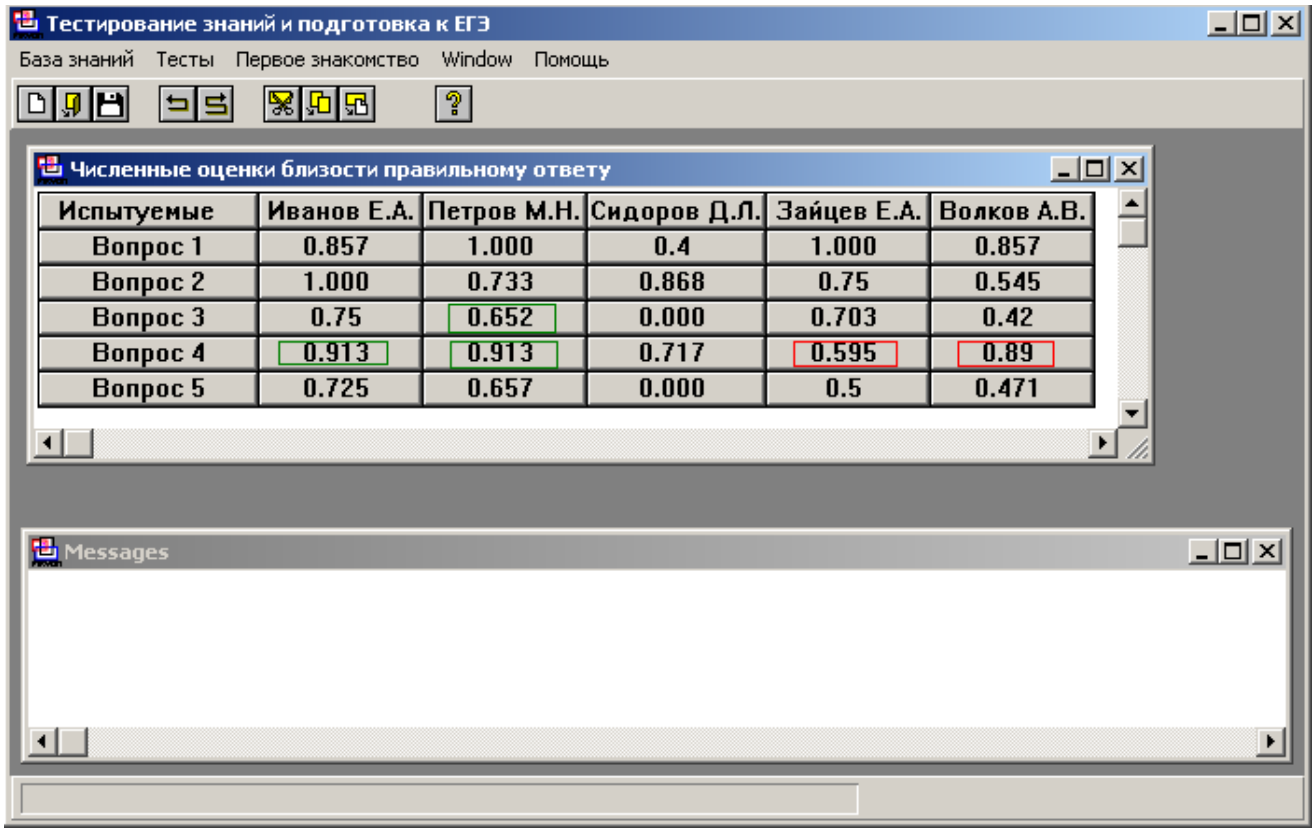


Рис. 6.33. Пример интерпретации ответа на тестовое задание открытой формы

На рис. 6.33 представлен интерфейс системы, а также интерпретация ответа на вопрос о влиянии переподгонки на частоту ошибок дерева принятия решений. Демо-версия системы (включая исходные тексты на языке Visual Prolog 5.2) доступна вместе с полным текстом диссертации в подразделе “Участник: Dmit-

ry.Mikhaylov” раздела “Страницы участников” профессионального информационно-аналитического ресурса www.machinelearning.ru, акты о результатах опытной эксплуатации приводятся в **приложении 2** диссертации. В представленном варианте реализованы следующие компоненты: формирование эталонов и базы лексико-синтаксических связей, соответствующих шестёркам (6.9), на основе СЯУ, тезаурус, подготовка и выполнение теста.



Испытуемые	Иванов Е.А.	Петров М.Н.	Сидоров Д.Л.	Зайцев Е.А.	Волков А.В.
Вопрос 1	0.857	1.000	0.4	1.000	0.857
Вопрос 2	1.000	0.733	0.868	0.75	0.545
Вопрос 3	0.75	0.652	0.000	0.703	0.42
Вопрос 4	0.913	0.913	0.717	0.595	0.89
Вопрос 5	0.725	0.657	0.000	0.5	0.471

Рис. 6.34. Результаты группового тестирования из примера на рис. 6.33 после автоматического согласования знаний о синонимии относительно различных СЯУ

Рис. 6.34 иллюстрирует применение *правила 6.1* к ситуациям языкового употребления, нашедшим отражение в тезаурусе. Каждая из уточнённых оценок близости правильным ответу обведена прямоугольником: зелёный – оценка стала выше, красный – оценка стала ниже первоначальной. Незначительное снижение оценок близости правильному ответу на *Вопрос 4* у испытуемых *Зайцева Е.А.* и *Волкова А.В.* обусловлено заменой выделенных ранее нулевых флексий у ряда слов, представленных в тезаурусе.

В целях более гибкой интерпретации ответа испытуемого оценки вида (5.5) в реализованном варианте программной системы вычисляются для случаев неполного ответа, орфографических ошибок, лишних слов, которые не фигурируют в лексико-синтаксических связях из представленных в базе знаний.

Рассмотрим более подробно каждый из трёх указанных случаев.

Случай 1. Неполный ответ – для всех слов и словосочетаний из ответа испытуемого нашлись прообразы в наиболее близком варианте правильного ответа, но для части слов правильного ответа не нашлось прообразов в ответе испытуемого.

Ненулевое значение оценки (5.6) будет только для тех из упущенных слов, которые в варианте правильного ответа являются синтаксически зависимыми по отношению к некоторым другим словам, присутствующим в анализируемом ответе. Здесь мы имеем обобщение оценки (5.6) на случай, когда для одного из сравниваемых объектов, а именно – основы того слова, которое упущено в ответе испытуемого, не определены признаки из множеств Mx_5 (указание на флексию зависимого слова), Mx_4 (сочетание флексий зависимого и главного слова), M_6 (указание на основу зависимого слова), M_7 (сочетание основы и флексии зависимого слова), M_8 (сочетание основ зависимого и главного слова). При этом один из сравниваемых объектов (тот, который соответствует основе упущенного слова) – фиктивный, а признаки из указанных множеств для этого объекта будут иметь значение “не определено”. Значение оценки (5.6) для упущенного слова здесь будет равно $-\log_2\left(1-\frac{2}{4}\right)\times\frac{3}{(8-3)+(8-3)+3}\approx 0.23$.

Случай 2. Орфографические ошибки (из допустимых) – слово из ответа испытуемого и слово из варианта правильного ответа являются различными формами одного и того же слова, допустимыми в рамках одной лексико-синтаксической связи (не обязательно в рамках рассматриваемой СЯУ). В этом случае оценка (5.6) для рассматриваемой пары слов вычисляется аналогично общему случаю, описанному в разделе 5.5.

Случай 3. “Лишние” слова. Здесь имеется в виду ситуация, когда все слова из наиболее близкого варианта правильного ответа нашли свой прообраз в ответе испытуемого, но в анализируемом ответе имеются слова, которые не нашли себе прообразов в правильном “варианте” (в том числе и на уровне словосочетаний). В этом случае ответ испытуемого не будет засчитан как неверный только тогда, когда “лишние” слова не фигурируют ни в одной лексико-синтаксической связи из представленных в базе знаний системы. При этом значение оценки (5.6) для каждого “лишнего” слова принимается равным нулю.

Допуская возможность объединения тестовых заданий в пакеты, в настоящей работе мы не затрагиваем общую оценку знаний тестируемого, которая формируется по результатам выполнения всех заданий пакета, а также построение самого сценария тестирования, который определяет состав пакета заданий. Указанные вопросы – тема отдельного исследования за рамками инженерных наук и относятся к соответствующим разделам педагогики и психологии.

Выводы

Таким образом, в шестой главе предложен метод компрессии текстовой базы знаний на основе выделения смысловых эталонов и последующего разделения предметных и языковых знаний.

Введением смыслового эталона на множестве СЭ-фраз достигается сокращение размера тезаурусной базы знаний для вычисления оценки схожести СЯУ при их независимом порождении не менее чем на 40–50%. При этом наибольший интерес для задач тестирования знаний представляет предложенный в разделе 6.2 метод выделения смыслового эталона на множестве СЭ-фраз с применением принципа формирования и кластеризации семантических отношений, разработанного автором и описанного в разделах 3.5 и 5.1.

Указанный метод выделения смыслового эталона позволяет в наибольшей степени учитывать особенности конкретного предметно-ограниченного подмно-

жества естественного языка. С другой стороны, при наличии существенных смысловых ограничений на перифразирование привлечение внешней программы синтаксического анализа, реализующей стратегию на основе наиболее вероятных связей, для разбора исходных СЭ-фраз позволяет выделить связи “объект-признак” в рамках формального контекста эталона с достаточно высокой точностью (менее 2% ошибок).

Вне зависимости от метода выделения смыслового эталона использование СЯУ в качестве единицы предварительного сжатия информации позволяет точно оценивать диапазоны значений требуемого объёма памяти для хранения текстов с учётом возможных видов синонимии.

Предложенная в настоящей главе концепция шаблона СЯУ может служить основой формирования синтаксических стратегий и правил относительно предметно-ограниченного подмножества естественного языка, в частности, в задаче формализации профессиональных знаний таксономическими структурами, рассмотренной в [98]. При построении тезауруса предметной области в виде формального контекста на основе совокупности шаблонов ситуаций языкового употребления сами шаблоны позволяют в автоматическом режиме выделять типы синтаксических отношений как классы формальных понятий. База синтаксических отношений, формируемых на основе шаблонов разных СЯУ по заданной предметной области, может быть использована при построении смысловых эталонов новых СЯУ.

Введённые в главе коэффициенты сжатия информации для формального контекста отдельной СЯУ и для теоретико-решеточного представления тезауруса как усреднённые показатели специфичности формальных понятий решётки могут быть использованы при автоматической сегментации решетки ФП тезауруса с применением алгоритма, предложенного Н. А. Степановой в [120], с целью выделения подмножеств заданного предметного языка.

ЗАКЛЮЧЕНИЕ

В заключении сформулируем основные научные и практические результаты настоящей диссертационной работы.

Основные научные результаты работы в области *разработки принципов и методов извлечения данных из текстов на естественном языке* состоят в следующем.

1. На основе теории *анализа формальных понятий* предложена *методика* автоматизированного формирования и экспериментальной оценки знаний, фиксируемых совокупностями классов семантической эквивалентности текстов в рамках ситуаций употребления естественного языка.

Новизной решения является *теоретико-решеточное представление* СЯУ в качестве информационной единицы тезауруса предметной области. За счёт использования формального понятия в качестве базового элемента информационного ресурса предложенное представление тезауруса решеткой формальных понятий позволяет оперировать данными на семантическом уровне без потери или недопустимого упрощения объектов и их признаков.

2. Сформулирован и теоретически обоснован *принцип* формирования и кластеризации семантических отношений на основе описаний ситуаций действительности множествами эквивалентных по смыслу фраз предметно-ограниченного подмножества естественного языка.

Новизна решения заключается в сравнении символьных последовательностей, составляющих *эквивалентные по смыслу* описания одного и того же *объекта* (ситуации) на заданном языке, с выделением изменяемых и неизменяемых частей для последующего анализа взаимного расположения фрагментов последовательностей в языковых конструкциях с разными логическими акцентами относительно одной и той же ситуации. Предложенная методика выявления закономерностей сосуществования словоформ в линейном ряду позволяет выделять для заданного естественного языка лучший способ выражения нужной мысли,

который составляет основу смыслового эталона. Сказанное актуально как для разработки стратегий и правил синтаксического анализа, так и для ролевой идентификации сущностей при формировании признаков сравниваемых текстов. Предложенный принцип формирования и кластеризации семантических отношений реализован в рамках демонстрационного варианта системы контроля знаний.

3. Разработаны *метод и алгоритмы* автоматизированного формирования *смыслового эталона* в виде *решётки формальных понятий*, а также *метод компрессии текстовой базы знаний* на основе выделенных эталонов.

Вне зависимости от пути формирования эталона его выделение сокращает размер базы знаний для оценки семантической схожести текстов предметно-ограниченного естественного языка текстов не менее чем на 40–50%.

В области *разработки и исследования методов и алгоритмов анализа текста* основной научный результат работы есть *метод численной оценки* семантической схожести *текстов* предметно-ограниченного естественного языка относительно ситуаций его употребления.

При этом *семантическая схожесть* текстов *оценивается* по *числу* признаков, которые характеризуют сочетаемость слов и разделяются *объектами* сравниваемых СЯУ относительно тезауруса, что немаловажно, в частности, при интерпретации результатов теста открытой формы в системах контроля знаний.

В области *разработки основ математической теории языков и грамматик* основной научный результат – это решение задачи построения системы целевых выводов в грамматике деревьев (Δ-грамматике).

В отличие от традиционных подходов к формализации преобразований помеченных деревьев, с целью нахождения последовательности преобразований с заданными свойствами автором *исследуется динамика функционирования совокупности правил Δ-грамматики в рамках её динамической информационной модели на базе ограниченных сетей Петри*. Такое решение учитывает недетерминированный характер порождения множества помеченных деревьев, а построение целевого вывода сводится к классическим задачам сетей Петри.

В качестве одного из наиболее значимых направлений развития полученных в работе результатов следует отметить этапы нормализации и комментирования текста при выполнении его предобработки для векторной модели, использующей матрицы “слово-контекст” или “пара-модель” [186]. При этом предложенное в работе теоретико-решёточное представление ситуации употребления естественного языка может служить основой выбора эвристики алгоритмом стемминга (определения основы либо корня для заданного исходного слова анализом его возможных морфологических форм), а также синтаксического разбора предложения на этапе комментирования. Ожидаемый здесь эффект – компромисс между точностью и полнотой информационного поиска как основными показателями эффективности работы поисковых систем.

СПИСОК ЛИТЕРАТУРЫ

1. *Аванесов В. С.* Композиция тестовых заданий: учебная книга для преподавателей вузов, учителей школ, аспирантов и студентов педвузов / В. С. Аванесов. М.: Адепт, 1998. 217 с.

2. АОТ: Автоматическая Обработка Текстов [Электронный ресурс]. Режим доступа: http://www.aot.ru/ (дата обращения: 23.06.2011).

3. *Апресян Ю. Д.* Избранные труды: в 2 т. Т. 1: Лексическая семантика. Синонимические средства языка / Ю. Д. Апресян. М.: Языки рус. культуры, 1995. 472 с.

4. *Биркгоф Г.* Теория решеток: пер. с англ. / Г. Биркгоф. М.: Наука, 1984. 566 с.

5. [Бродский А.] Алгоритмы контекстно-зависимого аннотирования Яндекса на РОМИП-2008 / А. Бродский, Р. Ковалев, М. Лебедев, Д. Лещинер, П. Сушин, И. Мучник // Труды РОМИП 2007-2008. СПб., 2008. С. 160–169.

6. *Вапник В. Н.* Восстановление зависимостей по эмпирическим данным / В. Н. Вапник. М.: Наука, 1979. 447 с.

7. *Вапник В. Н.* Теория распознавания образов. Статистические проблемы обучения / В. Н. Вапник, А. Я. Червоненкис. М.: Наука, 1974. 415 с.

8. [Васильев И. С.] Система автоматизированного контроля знаний на основе тестовых заданий открытой формы / И. С. Васильев, И. А. Кондратьев, Д. В. Михайлов, Г. М. Емельянов // XVIII научн. конф. преподавателей, аспирантов и студентов НовГУ: сб. тез. докл./ НовГУ им. Ярослава Мудрого. В. Новгород, 2011. С.50–51.

9. *Воронцов К. В.* Обзор современных исследований по проблеме качества обучения алгоритмов / К. В. Воронцов // Таврический вестн. информатики и математики. 2004. № 1. С. 5–24.

10. *Всеволодова А. В.* Компьютерная обработка лингвистических данных: учеб. пособие / А. В. Всеволодова. Ярославль: МУБиНТ, 2005. 67 с.

11. *Гаврилова Т. А.* Базы знаний интеллектуальных систем / Т. А. Гаврилова, В. Ф. Хорошевский. СПб.: Питер, 2001. 384 с.

12. *Гаскаров Д.В.* Интеллектуальные информационные системы: учеб. для вузов / Д. В. Гаскаров. М.: Высшая школа, 2003. 431 с.

13. *Герасимова И. А.* Формальная грамматика и интенсиональная логика / И. А. Герасимова. М.: Институт философии РАН, 2000. 156 с.

14. *Гладкий А. В.* Грамматики деревьев. I: Опыт формализации преобразований синтаксических структур естественного языка / А. В. Гладкий, И. А. Мельчук // Информационные вопросы семиотики, лингвистики и автоматического перевода. М., 1971. Вып. 1. С. 16–41.

15. *Гладкий А. В.* Грамматики деревьев. II: К построению Δ-грамматики для русского языка / А. В. Гладкий, И. А. Мельчук // Информационные вопросы семиотики, лингвистики и автоматического перевода. М., 1974. Вып. 4. С. 4–29.

16. [Гречников Е. А.] Поиск неестественных текстов / Е. А. Гречников, Г. Г. Гусев, А. А. Кустарев, А. М. Райгородский // Труды XI Всероссийской научной конференции RCDL’2009. Петрозаводск, 2009. С. 306–308.

17. [Гулин А.] Яндекс на РОМИП’2009. Оптимизация алгоритмов ранжирования методами машинного обучения [Электронный ресурс] / А. Гулин, П. Карпович, Д. Расковалов, И. Сегалович // Труды РОМИП’2009. Режим доступа: http://romip.ru/romip2009/15_yandex.pdf (дата обращения: 12.07.2012).

18. *Гусев В. Д.* Алгоритм выявления устойчивых словосочетаний с учетом их вариативности (морфологической и комбинаторной) [Электронный ресурс] / В. Д. Гусев, Н. В. Саломатина // Междунар. конф. по компьютерной лингвистике “Диалог-2004”. Режим доступа: http://www.dialog-21.ru/Archive/2004/Salomatina.htm (дата обращения: 23.06.2011).

19. *Гэри М.* Вычислительные машины и труднорешаемые задачи: пер. с англ. / М. Гэри, Д. Джонсон; под ред. А. А. Фридмана. М.: Мир, 1982. 416 с.

20. *Дейт К. Дж.* Введение в системы баз данных: пер. с англ. / К. Дж. Дейт. М.: Вильямс, 2001. 1071 с.

21. *Демьянков В. З.* Основы теории интерпретации и её приложения в вычислительной лингвистике / В. З. Демьянков. М.: Изд-во Моск. ун-та, 1985. 76 с.

22. *Демьянков В. З.* Специальные теории интерпретации в вычислительной лингвистике / В. З. Демьянков. М.: Изд-во Моск. ун-та, 1988. 87 с.

23. *Донской В. И.* Дискретные модели принятия решений при неполной информации / В. И. Донской, А. И. Башта. Симферополь: Таврия, 1992. 166 с.

24. *Дюкова Е.В.* Об алгоритме классификации на основе полного решающего дерева / Е. В. Дюкова, Н. В. Песков // 13-я Всерос. конф. “Математические методы распознавания образов” (ММРО-13): сб. докл. М., 2007. С. 125–126.

25. *Дюличева Ю. Ю.* О подходах к синтезу случайных и решающих лесов / Ю. Ю. Дюличева // 13-я Всерос. конф. “Математические методы распознавания образов” (ММРО-13): сб. докл. М., 2007. С. 126–127.

26. *Дюличева Ю. Ю.* Применение эмпирического решающего леса для фильтрации обучающих данных / Ю. Ю. Дюличева // Таврический вестн. информатики и математики. 2006. № 1. С. 55–61.

27. *Дюличева Ю. Ю.* Стратегии редукции решающих деревьев (обзор) / Ю. Ю. Дюличева // Таврический вестн. информатики и математики. 2002. № 1. С. 10–17.

28. *Евтушенко С. А.* Система анализа данных “Concept Explorer” / С. А. Евтушенко // Труды 7-ой национальной конференции по искусственному интеллекту КИИ-2000. М., 2000. С. 127–134.

29. *Емельянов Г. М.* Динамическая модель естественного языка в системах пользовательских интерфейсов / Г. М. Емельянов, Е. И. Зайцева, Д. В. Михайлов // Интеллектуализация обработки информации: тез. докл. Междунар. науч. конф. / Крымский научный центр НАН Украины, Таврический национальный университет. Симферополь, 2002. С. 120–121.

30. *Емельянов Г. М.* Динамическая модель естественного языка в системах пользовательских интерфейсов / Г. М. Емельянов, Д. В. Михайлов, Е. И. Зайцева

// Междунар. конф. по компьютерной лингвистике “Диалог-2002”. М.: Наука, 2002. Т. 2. С. 165–170.

31. [Емельянов Г.М.] К разработке распознающей системы анализа смысловых образов высказываний на естественном языке / Г.М. Емельянов, Е.И. Зайцева, Д. В. Михайлов, Е. П. Курашова // Распознавание образов и анализ изображений: новые информационные технологии (РОАИ-6-2002): труды 6-й Междунар. конф. / НовГУ им. Ярослава Мудрого. В. Новгород, 2002. Т. 1. С. 220–223.

32. *Емельянов Г. М.* Концептуально-ситуационное моделирование процесса перифразирования высказываний Естественного Языка как обучение на основе прецедентов / Г. М. Емельянов, А. Н. Корнышов, Д. В. Михайлов // Интеллектуализация обработки информации: тез. докл. Междунар. науч. конф. / Крымский научный центр НАН Украины. Симферополь, 2006. С. 78–79.

33. *Емельянов Г. М.* Концептуально-ситуационное моделирование процесса перифразирования высказываний Естественного Языка как обучение на основе прецедентов / Г. М. Емельянов, А. Н. Корнышов, Д. В. Михайлов // Искусственный интеллект. 2006. № 2. С. 72–75.

34. *Емельянов Г. М.* Применение реляционной модели представления данных для организации словаря в системе анализа семантической эквивалентности текстов естественного языка [Электронный ресурс] / Г. М. Емельянов, Д. В. Михайлов, Д. В. Силанов // Ученые записки Новгородского ун-та. Режим доступа: http://admin.novsu.ac.ru/uni/scpapers.nsf/publications (дата обращения: 23.06.2011).

35. *Емельянов Г. М.* Синонимические преобразования в задаче анализа эквивалентности смысловых образов высказываний на уровне сверхфразовых единств / Г. М. Емельянов, Д. В. Михайлов, Е. И. Зайцева // Распознавание образов и анализ изображений: новые информационные технологии (РОАИ-6-2002): труды 6-й Междунар. конф. / НовГУ им. Ярослава Мудрого. В. Новгород, 2002. Т. 1. С. 215–219.

36. *Журавлёв Ю. И.* “Распознавание”. Математические методы. Программная система. Практические применения / Ю. И. Журавлёв, В. В. Рязанов, О. В. Сенько. М.: ФАЗИС, 2006. 176 с.

37. *Журавлёв Ю. И.* Об алгебраическом подходе к решению задач распознавания или классификации / Ю. И. Журавлёв // Проблемы кибернетики. М., 1978. Вып. 33. С. 5–68.

38. [Журавлёв Ю. И.] Система распознавания интеллектуальных заимствований “Антиплагиат” / Ю. И. Журавлёв, К. В. Рудаков, А. С. Инякин, А. А. Кирсанов, А. В. Лисица, Г. В. Никитов, Н. В. Песков, М. Ю. Романов, Ю. В. Чехович, Р. И. Яминов // 12-я Всерос. конф. “Математические методы распознавания образов” (ММРО-12): сб. докл. М., 2005. С. 329–332.

39. *Заболеева-Зотова А.В.* Лингвистическое обеспечение автоматизированных систем: учеб. пособие / А. В. Заболеева-Зотова, В. А. Камаев. М.: Высшая школа, 2008. 244 с.

40. *Загоруйко Н. Г.* Прикладные методы анализа данных и знаний / Н. Г. Загоруйко. Новосибирск: изд-во ин-та математики, 1999. 270 с.

41. *Залешин М. В.* Формирование и кластеризация синтаксических отношений в текстах предметного языка / М. В. Залешин, Д. В. Михайлов // XVII научн. конф. преподавателей, аспирантов и студентов НовГУ: сб. тез. докл./ НовГУ им. Ярослава Мудрого. Великий Новгород, 2010. Ч.3. С.17.

42. *Игнатов Д. И.* О поиске сходства Интернет-документов с помощью частых замкнутых множеств признаков / Д. И. Игнатов, С. О. Кузнецов // Труды 10-ой национальной конференции по искусственному интеллекту КИИ-2006. М., 2006. Т. 2. С. 249–258.

43. Искусственный интеллект: в 3 кн. / под ред. Э. В. Попова. М.: Радио и связь, 1990.

44. *Караулов Ю. Н.* Лингвистическое конструирование и тезаурус литературного языка / Ю. Н. Караулов. М.: Наука, 1981. 366 с.

45. *Кибрик А. Е.* Очерки по общим и прикладным вопросам языкознания / А. Е. Кибрик. М.: КомКнига, 2005. 332 с.

46. *Кондратов А. М.* Звуки и знаки / А. М. Кондратов. М.: Знание, 1978. 208 с.

47. *Корнышов А. Н.* Иерархизация системы предикатов семантических отношений / А. Н. Корнышов, Д. В. Михайлов // Интеллектуализация обработки информации: тез. докл. Междунар. науч. конф. / Крымский научный центр НАН Украины. Симферополь, 2008. С. 130–131.

48. *Корнышов А. Н.* Концептуально-ситуационное моделирование высказываний естественного языка в задаче анализа их смысловой эквивалентности / А. Н. Корнышов, Д. В. Михайлов // Вестн. Новгородского гос. ун-та им. Ярослава Мудрого, сер. “Техн. науки”. 2005. № 34. С. 76–80.

49. *Корнышов А. Н.* Концептуальный уровень и его использование в задаче моделирования синонимических преобразований высказываний естественного языка / А. Н. Корнышов, Д. В. Михайлов // Математика в вузе: материалы XVIII Междунар. науч.-метод. конф. / Петербургский гос. ун-т путей сообщения. СПб., 2005. С. 118–120.

50. *Корнышов А. Н.* Обучение на основе прецедентов в задаче распознавания смысловой эквивалентности / А. Н. Корнышов, Д. В. Михайлов // XIII науч. конф. преподавателей, аспирантов и студентов НовГУ: сб. тез. докл. / НовГУ им. Ярослава Мудрого. Великий Новгород, 2006. С. 136.

51. *Корнышов А. Н.* Предикаты семантических отношений в задаче моделирования системы концептуальных зависимостей в тезаурусе предметной области / А. Н. Корнышов, Д. В. Михайлов // XIV науч. конф. преподавателей, аспирантов и студентов НовГУ: сб. тез. докл. / НовГУ им. Ярослава Мудрого. Великий Новгород, 2007. С. 182–183.

52. *Корнышов А.Н.* Таксономия знаний в задаче распознавания семантических отношений / А. Н. Корнышов, Д. В. Михайлов // Распознавание-2008: сб. материалов VIII Междунар. конф. / Курск. гос. техн. ун-т. Курск, 2008. Ч. 1. С. 183–185.

53. *Котов В. Е.* Сети Петри / В. Е. Котов. М.: Наука, 1984. 160 с.

54. *Красильникова В. А.* Подготовка заданий для компьютерного тестирования: метод. рекомендации / В. А. Красильникова. Оренбург: ИПК ГОУ ОГУ, 2004. 31 с.

55. *Кубрякова Е. С.* Язык и знание: На пути получения знаний о языке: части речи с когнитивной точки зрения. Роль языка в познании мира / Е. С. Кубрякова. М.: Языки славянской культуры, 2004. 555 с.

56. *Леонтьева Н. Н.* “Политекст”: информационный анализ политических текстов / Н. Н. Леонтьева // Научно-техническая информация. М.: ВИНИТИ, 1995. № 4. Сер. 2. С. 20-24.

57. *Леонтьева Н. Н.* О методах смысловой компрессии текста [Электронный ресурс] / Н. Н. Леонтьева // X Всерос. объединенная конф. “Интернет и современное общество” (IMS-2007). Режим доступа: http://www.ict.edu.ru/vconf/files/7881.pdf (дата обращения: 23.06.2011).

58. *Леонтьева Н. Н.* Русский общесемантический словарь (РОСС): структура, наполнение / Н. Н. Леонтьева // Научно-техническая информация. М.: ВИНИТИ, 1997. № 12. Сер. 2. С. 5–20.

59. *Ломазова И. А.* Вложенные сети Петри: моделирование и анализ распределенных систем с объектной структурой / И. А. Ломазова. М.: Научный мир, 2004. 208 с.

60. *Майоров А. Н.* Теория и практика создания тестов для системы образования. Как выбирать, создавать и использовать тесты для целей образования / А. Н. Майоров. М.: Нар. образование, 2000. 351 с.

61. *Мельников Г. П.* Системная типология языков: Принципы, методы, модели / Г. П. Мельников. М.: Наука, 2003. 395 с.

62. *Мельчук И.А.* Опыт теории лингвистических моделей “Смысл⇔Текст”: Семантика, синтаксис / И. А. Мельчук. М.:Языки рус. культуры, 1999. 345 с.

63. *Михайлов Д. В.* Автоматизация накопления знаний о синонимии текстов предметного языка / Д. В. Михайлов, Г. М. Емельянов // Распознавание-2010: сб. материалов IX Междунар. конф. / Курск. гос. техн. ун-т. Курск, 2010. С. 186–188.

64. *Михайлов Д. В.* Анализ формальных понятий и сжатие текстовой информации в задаче автоматизированного контроля знаний / Г. М. Емельянов, Д. В. Михайлов // 15-я Всерос. конф. “Математические методы распознавания образов” (ММРО-15): сб. докл. М., 2011. С. 581–584.

65. *Михайлов Д. В.* Вопросы использования предметных и естественных языков в задачах открытого тестирования / Д. В. Михайлов // Великий Новгород – город университетский: материалы юбилейной науч.-практ. конф. / НовГУ им. Ярослава Мудрого. Великий Новгород, 2003. С. 103–104.

66. *Михайлов Д. В.* Вопросы моделирования семантической связанности для систем понимания текста / Г. М. Емельянов, Д. В. Михайлов // Распознавание-2001: сб. материалов 5-й Междунар. конф. / Курский гуманитарно-техн. инст-т; Курский гос. техн. ун-т. Курск, 2001. Ч. 1. С. 56–58.

67. *Михайлов Д. В.* Вопросы моделирования семантической связанности для систем автоматизированного тестирования знаний / Г. М. Емельянов, Д. В. Михайлов // Докл. X Всерос. конф. “Математические методы распознавания образов” (ММРО-10). М., 2001. С. 53–56.

68. *Михайлов Д. В.* Вопросы построения механизма суммирования смысла для систем распознавания текстов на естественном языке / Г. М. Емельянов, Д. В. Михайлов // Методы и средства обработки сложной графической информации: тез. докл. VI Всерос. конф. с участием стран СНГ / НИИ прикладной математики и кибернетики ННГУ. Нижний Новгород, 2001. С. 83–85.

69. *Михайлов Д. В.* Иерархия семантических отношений в задаче построения Модели Управления предикатного слова / Д. В. Михайлов, Г. М. Емельянов // Распознавание-2005: сб. материалов 7-й Междунар. конф. / Курский гос. техн. ун-т. Курск, 2005. С. 42–43.

70. *Михайлов Д. В.* Информационное наполнение дерева в задаче исследования динамики функционирования Δ-грамматики / Д. В. Михайлов, Г. М. Емельянов // Распознавание-2003: сб. материалов 6-й Междунар. конф. / Курский гос. техн. ун-т. Курск, 2003. Ч. 1. С. 35–37.

71. *Михайлов Д. В.* Информационно-логическая модель системы правил Δ-грамматики / Д. В. Михайлов, Г. М. Емельянов // Известия СПбГЭТУ “ЛЭТИ”, сер. “Информатика, управление и компьютерные технологии”. СПб., 2003. Вып. 3. С. 96–102.

72. *Михайлов Д. В.* К вопросу автоматизации пополнения базы данных лексических функций в задаче установления смысловой эквивалентности текстов естественного языка / Д. В. Михайлов, Г. М. Емельянов // Вестн. Новгородского гос. ун-та им. Ярослава Мудрого, сер. “Техн. науки”. 2007. № 44. С. 45–49.

73. *Михайлов Д. В.* Кластеризация семантических знаний в задаче распознавания ситуаций смысловой эквивалентности / Д. В. Михайлов, Г. М. Емельянов // 13-я Всерос. конф. “Математические методы распознавания образов” (ММРО-13): сб. докл. М., 2007. С. 500–503.

74. *Михайлов Д. В.* Модель сортовой системы языка в задаче построения семантического образа высказывания на уровне глубинного синтаксиса / Д. В. Михайлов, Г. М. Емельянов // Таврический вестн. информатики и математики. 2006. № 1. С. 79–90.

75. *Михайлов Д. В.* Модель сортовой системы языка в задаче построения семантического образа высказывания на уровне глубинного синтаксиса / Д. В. Михайлов, Г. М. Емельянов // Интеллектуализация обработки информации: тез. докл. Междунар. науч. конф. / Крымский научный центр НАН Украины. Симферополь, 2006. С. 148–150.

76. *Михайлов Д. В.* Морфология и синтаксис в задаче семантической кластеризации / Д. В. Михайлов, Г. М. Емельянов // 14-я Всерос. конф. “Математические методы распознавания образов” (ММРО-14): сб. докл. М., 2009. С. 563–566.

77. *Михайлов Д. В.* Пополнение словаря моделей управления в задаче анализа семантической эквивалентности текстовых документов / Д. В. Михайлов, Г. М. Емельянов // Методы и средства обработки сложной графической информа-

ции: тез. докл. VIII Всерос. науч. конф. / ГНУ “НИИ ПМК ННГУ”. Нижний Новгород, 2005. С. 88–93.

78. *Михайлов Д. В.* Построение динамической модели естественного языка применительно к разработке языковой базы знаний / Г. М. Емельянов, Е. И. Зайцева, Д. В. Михайлов // Искусственный интеллект. 2002. № 2. С. 443–446.

79. *Михайлов Д. В.* Построение модели объекта информационного пространства применительно к исследованию динамики функционирования Δ-грамматик / Д. В. Михайлов, Г. М. Емельянов // Вестн. Новгородского гос. ун-та им. Ярослава Мудрого, сер. “Техн. науки”. 2004. № 26. С. 131–136.

80. *Михайлов Д. В.* Построение модели управления предикатного слова на основе его лексикографического толкования / Г. М. Емельянов, Д. В. Михайлов // Таврический вестн. информатики и математики. 2005. № 1. С. 35–48.

81. *Михайлов Д. В.* Представление смысла в задаче установления семантической эквивалентности высказываний / Д. В. Михайлов, Г. М. Емельянов // Вестн. Новгородского гос. ун-та им. Ярослава Мудрого, сер. “Техн. науки”. 2004. № 28. С. 106–110.

82. *Михайлов Д. В.* Применение аппарата ограниченных сетей Петри для построения динамической модели естественного языка / Г. М. Емельянов, Е. И. Зайцева, Д. В. Михайлов // Интеллектуализация обработки информации: тез. докл. Междунар. науч. конф. / Крымский научный центр НАН Украины, Таврический национальный университет. Симферополь, 2002. С. 121–122.

83. *Михайлов Д. В.* Применение семантических полей словаря РОСС в задаче построения модели управления предикатного слова / Д. В. Михайлов, Г. М. Емельянов // 12-я Всерос. конф. “Математические методы распознавания образов” (ММРО-12): сб. докл. М., 2005. С. 382–385.

84. *Михайлов Д. В.* Распознавание сверхфразовых единств при установлении эквивалентности смысловых образов высказываний в общей задаче моделирования языковой деятельности / Г. М. Емельянов, Д. В. Михайлов //

Известия СПбГЭТУ “ЛЭТИ”, сер. “Информатика, управление и компьютерные технологии”. СПб., 2003. Вып. 1. С. 65–73.

85. *Михайлов Д. В.* Семантическая кластеризация текстов предметных языков (морфология и синтаксис) / Д. В. Михайлов, Г. М. Емельянов // Компьютерная оптика. 2009. Т. 33, № 4. С. 473–480.

86. *Михайлов Д. В.* Семантическая схожесть текстов в задаче автоматизированного контроля знаний / Д. В. Михайлов, Г. М. Емельянов // 8-я Междунар. конф. “Интеллектуализация обработки информации” (ИОИ-2010): сб. докл. М., 2010. С. 516–519.

87. *Михайлов Д. В.* Теоретические основы построения открытых вопросно-ответных систем. Семантическая эквивалентность текстов и модели их распознавания: монография / Д. В. Михайлов, Г. М. Емельянов; НовГУ им. Ярослава Мудрого. Великий Новгород, 2010. 286 с.

88. *Михайлов Д. В.* Установление смысловой эквивалентности высказываний: на пути к решению проблемы / Г. М. Емельянов, Д. В. Михайлов // Искусственный интеллект. 2004. № 2. С. 86–90.

89. *Михайлов Д. В.* Установление смысловой эквивалентности высказываний: на пути к решению проблемы / Г. М. Емельянов, Д. В. Михайлов // Интеллектуализация обработки информации: тез. докл. Междунар. науч. конф. / Крымский научный центр НАН Украины. Симферополь, 2004. С. 70.

90. *Михайлов Д. В.* Формирование и кластеризация знаний о синонимии в рамках стандартных лексических функций / Д. В. Михайлов, Г. М. Емельянов // Сб. науч. статей / НовГУ им. Ярослава Мудрого. В. Новгород, 2009. С. 17–33.

91. *Михайлов Д. В.* Формирование и кластеризация контекстов для существительных русского языка в рамках конверсивных замен / Д. В. Михайлов, Н. А. Степанова, И. И. Юрченко // Физика и механика материалов: прил. к науч.-теорет. и прикл. журн. “Вестник Новгородского государственного университета имени Ярослава Мудрого”. 2009. № 50. С. 31–34.

92. *Михайлов Д. В.* Формирование и кластеризация понятий в задаче автоматизированного построения тезауруса предметной области / Д. В. Михайлов, Г. М. Емельянов // Распознавание-2008: сб. материалов VIII Междунар. конф. / Курский гос. техн. ун-т. Курск, 2008. Ч. 2. С. 20–22.

93. *Михайлов Д. В.* Формирование и кластеризация понятий на основе множества ситуационных контекстов / Д. В. Михайлов, Г. М. Емельянов, Н. А. Степанова // Таврический вестн. информатики и математики. 2008. № 2. С. 79–88.

94. *Михайлов Д. В.* Формирование и кластеризация понятий на основе множества ситуационных контекстов / Д. В. Михайлов, Г. М. Емельянов, Н. А. Степанова // Интеллектуализация обработки информации: тез. докл. Междунар. науч. конф. / Крымский науч. центр НАН Украины. Симферополь, 2008. С. 168–170.

95. *Михайлов Д.В.* Смысловые эталоны в моделях распознавания и компрессии текстов / Д.В. Михайлов // Вестн. Новгородского гос. ун-та им. Ярослава Мудрого. 2012. № 68 (в печати).

96. *Михайлов Д.В.* Формирование смысловых эталонов и интерпретация результатов открытых тестов в системах контроля знаний / Д.В. Михайлов // Вестн. Новгородского гос. ун-та им. Ярослава Мудрого, сер. “Техн. науки”. 2011. № 65. С. 83–87.

97. Моделирование языковой деятельности в интеллектуальных системах / под ред. А. Е. Кибрика и А. С. Нариньяни. М.: Наука, 1987. 279 с.

98. *Мошков И. С.* Компьютерная система анализа текста таксономического типа применительно к оценке профессиональных знаний / А. Н. Краснов, И. С. Мошков, В. Н. Якимов // Междунар. науч.-практ. конф. “Инновация-2011”: сб. науч. статей / Ташкентский гос. техн. ун-т. Ташкент, 2011. С. 287–289.

99. *Налимов В. В.* Вероятностная модель языка. О соотношении естественных и искусственных языков / В. В. Налимов. М.: Наука, 1974. 272 с.

100. *Нариньяни А. С.* Кентавр по имени ТЕОН: тезаурус+онтология / А. С. Нариньяни // Междунар. конф. по компьютерной лингвистике “Диалог-2001”: труды конф. М., 2001. Т. 1. С. 184–188.

101. *Осипов Г. С.* Приобретение знаний интеллектуальными системами: основы теории и технологии / Г. С. Осипов. М.: Наука, 1997. 112 с.

102. *Останин К.С.* Система компьютерного тестирования “ТестЭкзаменатор” [Электронный ресурс] / К. С. Останин // Междунар. конгр. конференций “Информационные Технологии в Образовании” (ИТО-2003). Режим доступа: http://www.bitpro.ru/ito/2003/VI/VI-0-2562.html (дата обращения: 24.06.2011).

103. *Павиленис Р. И.* Проблема смысла: современный логико-философский анализ языка / Р. И. Павиленис. М.: Мысль, 1983. 286 с.

104. *Питерсон Дж.* Теория сетей Петри и моделирование систем: пер. с англ. / Дж. Питерсон. М.: Мир, 1984. 264 с.

105. *Позин П. А.* Сравнительный анализ открытого и закрытого ответа на тестовое задание / П. А. Позин, В. Д. Синявский // Развитие системы тестирования в России: тез. докл. III Всерос. науч.-метод. конф. / под ред. Л. С. Гребнева; Центр тестирования Министерства образования РФ. М., 2001. С. 207.

106. *Попов Э. В.* Общение с ЭВМ на естественном языке / Э. В. Попов. М.: Наука, 1982. 360 с.

107. *Поспелов Д. А.* Ситуационное управление: теория и практика / Д. А. Поспелов. М.: Наука, 1986. 288 с.

108. *Потапов А.С.* Распознавание образов и машинное восприятие: общий подход на основе принципа минимальной длины описания / А. С. Потапов. СПб.: Политехника, 2007. 548 с.

109. Представление знаний в человеко-машинных и робототехнических системах: в 4 т. // Отчет РГ-18 КНВВТ. М.: ВЦ АН СССР: ВИНИТИ, 1984.

110. Программный пакет синтаксического разбора и машинного перевода [Электронный ресурс]. Режим доступа: http://cs.isa.ru:10000/dwarf/ (дата обращения: 18.11.2009).

111. *Рубашкин В. Ш.* Представление и анализ смысла в интеллектуальных системах / В. Ш. Рубашкин. М.: Наука, 1989. 192 с.

112. *Рыков В. В.* Корпус текстов как семиотическая система и онтология речевой деятельности [Электронный ресурс] / В. В. Рыков // Междунар. конф. по компьютерной лингвистике “Диалог-2004”. Режим доступа: http://www.dialog-21.ru/Archive/2004/Rykov.htm (дата обращения: 23.06.2011).

113. Свидетельство об офиц. регистрации прогр. для ЭВМ № 2010617263. Программа формирования синтаксических отношений на множестве семантически эквивалентных фраз / Залешин М. В., Михайлов Д. В., Емельянов Г. М.; заявитель и правообладатель “Новгородский государственный университет имени Ярослава Мудрого”. Заявка № 2010615398; заявл. 02.09.10.; зарег. 29.10.10.

114. *Севбо И. П.* Структура связного текста и автоматизация реферирования / И. П. Севбо. М.: Наука, 1969. 135 с.

115. *Силанов Д. В.* Применение теорий Лексических Значений слов при распознавании ситуаций смысловой эквивалентности / Д. В. Силанов, Д. В. Михайлов // XIV науч. конф. преподавателей, аспирантов и студентов НовГУ: сб. тез. докл. / НовГУ им. Ярослава Мудрого. Великий Новгород, 2007. С. 183–184.

116. Системы искусственного интеллекта. Практический курс: учеб. пособие / под ред. И. Ф. Астаховой. М.: БИНОМ. Лаборатория знаний, 2008. 292 с.

117. *Смирнова Е. И.* Моделирование структуры состояний сложной системы для задач прогнозирования / Е. И. Смирнова // Искусственный интеллект. 2000. № 2. С. 196–199.

118. *Солганик Г. Я.* Стилистика текста: учеб. пособие / Г. Я. Солганик. М.: Флинта, Наука, 1997. 253 с.

119. *Соснин П. И.* Человеко-компьютерная диалогика / П. И. Соснин. Ульяновск: УлГТУ, 2001. 285 с.

120. *Степанова Н. А.* Формирование и кластеризация понятий в задаче распознавания образов в пространстве знаний / Н. А. Степанова, Г. М. Емельянов // 13-я Всерос. конф. “Математические методы распознавания образов” (ММРО-13). М., 2007. С. 206–209.

121. *Тестелец Я. Г.* Введение в общий синтаксис / Я. Г. Тестелец. М.: РГГУ, 2001. 800 с.

122. *Тихомиров И. А.* Интеграция лингвистических и статистических методов поиска в поисковой машине “Exactus” [Электронный ресурс] / И. А. Тихомиров, И. В. Смирнов // Междунар. конф. по компьютерной лингвистике “Диалог-2008”. Режим доступа: http://www.dialog-21.ru/dialog2008/materials/html/80.htm (дата обращения: 23.06.2011).

123. *Тузов В. А.* Математическая модель языка / В. А. Тузов. Л.: Изд-во Ленингр. ун-та, 1984. 176 с.

124. *Фомичев В. А.* Математические основы представления смысла текстов для разработки лингвистических информационных технологий / В. А. Фомичев // Информационные технологии. 2002. № 10. С. 16–25; № 11. С. 34–45.

125. *Фомичев В. А.* Формализация проектирования лингвистических процессоров / В. А. Фомичев. М.: Макс Пресс, 2005. 367 с.

126. *Хант Э.* Искусственный интеллект: пер. с англ. / Э. Хант. М.: Мир, 1978. 558 с.

127. *Хомский Н.* Формальные свойства грамматик / Н. Хомский // Кибернетический сборник. М., 1961. № 2. С. 121–130.

128. *Хомский Н.* Язык и мышление: пер. с англ. / Н. Хомский. М.: изд-во Моск. ун-та, 1972. 122 с.

129. *Челышкова М. Б.* Теория и практика конструирования педагогических тестов: учеб. пособие / М. Б. Челышкова. М.: Логос, 2002. 432 с.

130. *Юрченко И.И.* Программный комплекс вычисления частотных характеристик глаголов для задачи формирования и кластеризации понятий / И. И. Юрченко, Д. В. Михайлов // XV науч. конф. преподавателей, аспирантов и студентов НовГУ: сб. тез. докл. / НовГУ им. Ярослава Мудрого. В. Новгород, 2008. С. 245.

131. *Юрченко И. И.* Семантическая кластеризация текстов русского языка / И. И. Юрченко, Д. В. Михайлов // XVI науч. конф. преподавателей, аспирантов и

студентов НовГУ: сб. тез. докл. / НовГУ им. Ярослава Мудрого. Великий Новгород, 2009. Ч. 3. С. 34–35.

132. Яндекс. Словари [Электронный ресурс]. Режим доступа: http://slovari.yandex.ru (дата обращения: 23.06.2011).

133. *Ясницкий Л.Н.* Введение в искусственный интеллект: учеб. пособие для студ. высш. учеб. заведений / Л. Н. Ясницкий. М.: Академия, 2008. 176 с.

134. *Antonova Alexandra.* Building a Web-based parallel corpus and filtering out machine-translated text [Электронный ресурс] / Alexandra Antonova, Alexey Misyurev // Proceedings of the 4th Workshop on Building and Using Comparable Corpora: Comparable Corpora and the Web. Режим доступа: http://aclweb.org/anthology-new/W/W11/W11-12.pdf (дата обращения: 12.07.2012).

135. [Beloozerov V. N.] Construction and Use of a Thesaurus in Image Analysis and Processing / V. N. Beloozerov, I. B. Gurevich, D. M. Murashov, Yu. O. Trusova // Pattern Recognition and Image Analysis. 2003. Vol. 13, N 1. P. 67–69.

136. *Beloozerov V. N.* Representation of the Ontology of an Image Analysis Domain for Optimization of Information Retrieval / V. N. Beloozerov, I. B. Gurevich, Yu. O. Trusova // Pattern Recognition and Image Analysis. 2005. Vol. 15, N 2. P. 358–360.

137. [Beloozerov V. N.] Searching for Solutions in the Image Analysis and Processing Knowledge Base / V. N. Beloozerov, D. M. Murashov, Yu. O. Trusova, D. A. Yanchenko // Pattern Recognition and Image Analysis. 2005. Vol. 15, N 2. P. 361–364.

138. [Beloozerov V. N.] Thesaurus for Image Analysis: Basic Version / V. N. Beloozerov, I. B. Gurevich, N. G. Gurevich, D. M. Murashov, Yu. O. Trusova // Pattern Recognition and Image Analysis. 2003. Vol. 13, N 4. P. 556–569.

139. *Borschev Vladimir.* Genitives, Types and Sorts: the Russian Genitive of Measure [Электронный ресурс] / Vladimir Borschev, Barbara H. Partee. Режим доступа: http://semanticsarchive.net/Archive/GJlMzYwN/B&P_PossWkshp04.pdf (дата обращения: 25.06.2011).

140. *Burmeister Peter*. Formal Concept Analysis with ConImp: introduction to the Basic Features [Электронный ресурс] / Peter Burmeister. Режим доступа: http://www.mathematik.tu-darmstadt.de/~burmeister/ConImpIntro.pdf (дата обращения: 24.06.2011).

141. *Carpineto Claudio.* Concept Data Analysis: theory and Applications / Claudio Carpineto, Giovanni Romano. Chichester: Wiley, 2004. 220 c.

142. [Colantonio S.] Cell Image Analysis Ontology / S. Colantonio, I. Gurevich, M. Martinelli, O. Salvetti, Yu. Trusova // Pattern Recognition and Image Analysis. 2008. Vol. 18, N 2. P. 332–341.

143. *Emel'yanov G. M.* Analysis of Semantic Relations in Classification of Sense Images of Statements / G. M. Emel'yanov, D. V. Mikhailov, N. A. Stepanova // Pattern Recognition and Image Analysis. 2007. Vol. 17, N 2. P. 274–278.

144. *Emelyanov G. M.* Algebra of the Logical Simulation of Hypersegment Image Databases / G. M. Emelyanov, E. I. Smirnova // Pattern Recognition and Image Analysis. 2000. Vol. 10, N 1. P. 156–163.

145. *Emelyanov G. M.* Application of the computer thesaurus for automation of updating of the Government Patterns's dictionary / G. M. Emelyanov, D. V. Mikhailov, N. A. Stepanova // VI International Congress on Mathematical Modeling. Book of Abstracts / University of Nizhny Novgorod. Nizhny Novgorod, 2004. P. 352.

146. *Emelyanov G. M.* Development of Recognition System of Analysis of Semantic Images of Natural Language Statements / G. M. Emelyanov, E. I. Zaitseva, D. V. Mikhailov, E. P. Kurashova // Pattern Recognition and Image Analysis. 2003. Vol. 13, N 2. P. 251–253.

147. *Emelyanov G. M.* Logical Model Of Hypertext Image Database / G. M. Emelyanov, E. I. Smirnova // Pattern Recognition and Image Analysis. 1999. Vol. 9, N 3. P. 458–491.

148. *Emelyanov G. M.* Semantic Analysis in Computer-Aided Systems of Speech Understanding / G. M. Emelyanov, T. V. Krechetova, E. P. Kurashova // Pattern Recognition and Image Analysis. 1998. Vol. 8, N 3. P. 408–410.

149. *Emelyanov G. M.* Semantic relation analysis for classification of meaning pattern of utterances / G. M. Emelyanov, D. V. Mikhailov, N. A. Stepanova // 7th Int. Conf. on Pattern Recognition and Image Analysis: new Information Technologies (PRIA-7-2004). Conf. Proc. / SPbETU. St. Petersburg, 2004. Vol. II. P. 460–461.

150. *Emelyanov G. M.* Semantic Relation Analysis for Classification of the Meaning Patterns of Utterances / G. M. Emelyanov, D. V. Mikhailov, N. A. Stepanova // Pattern Recognition and Image Analysis. 2005. Vol. 15, N 2. P. 382–383.

151. *Emelyanov G. M.* Tree Grammars in the Problems of Searching for Images by Their Verbal Descriptions / G. M. Emelyanov, T. V. Krechetova, E. P. Kurashova // Pattern Recognition and Image Analysis. 2000. Vol. 10, N 4. P. 520–526.

152. *Fomichov Vladimir A.* Theory of K-Calculuses as a Powerful and Flexible Mathematical Framework for Building Ontologies and Designing Natural Language Processing Systems / Vladimir A. Fomichov // 5th International Conference FQAS 2002. Berlin: Springer-Verlag, 2002. P. 183–196.

153. *Ganter B.* Formal Concept Analysis – Mathematical Foundations / B. Ganter, R. Wille. Berlin: Springer-Verlag, 1999. 284 c.

154. [Ganter B.] Formal Concept Analysis: Foundations and Applications / B. Ganter, G. Stumme, R. Wille [eds.]. Berlin: Springer-Verlag, 2005. 349 c.

155. [Gurevich I. B.] An Open General-Purposes Research System for Automating the Development and Application of Information Technologies in the Area of Image Processing, Analysis, and Evaluation / I. B. Gurevich, A. V. Khilkov, I. V. Koryabkina, D. M. Murashov, Yu. O. Trusova // Pattern Recognition and Image Analysis. 2006. Vol. 16, N 4. P. 530–563.

156. [Haan B. J.] IRIS: Hipermedia Services / B. J. Haan, P. Kahn, V. A. Riley, J. H. Coombs, N. K. Meyrowitz // Communication of the ACM. 1992. Vol. 36, N 1. P. 36–51.

157. *Hastie T.* The Elements of Statistical Learning: Data Mining, Inference, and Prediction / T. Hastie, R. Tibshirani, J. Friedman. Berlin: Springer-Verlag, 2009. 746 c.

158. *Kuznetsov S. O.* Comparing Performance of Algorithms for Generating Concept Lattices / S. O. Kuznetsov, S. A. Obiedkov // Journal of Experimental and Theoretical Artificial Intelligence. 2002. Vol. 14. P. 189–216.

159. *Kuznetsov S. O.* On stability of a formal concept / S. O. Kuznetsov // Annals of Mathematics and Artificial Intelligence. 2007. Vol. 49. P. 101–115.

160. *Lapshov Y. A.* Human Interruptions Management in Corporate Modeling Environment WIQA.NET / Y. A. Lapshov // Interactive Systems And Technologies: The Problems of Human-Computer Interaction. Collections of scientific papers / ULSTU. Ulyanovsk, 2009. Vol. III. P. 229–234.

161. *Maklaev V. A.* Tool and Technological Environment for Generation and Use of Design Organization Experience / V. A. Maklaev, P. I. Sosnin // Interactive Systems And Technologies: The Problems of Human-Computer Interaction. Collections of scientific papers / ULSTU. Ulyanovsk, 2009. Vol. III. P. 120–126.

162. *Mel'cuk Igor A.* Explanatory Combinatorial Dictionary of Modern Russian. Semantico-Syntactic Studies of Russian Vocabulary / Igor A. Mel'cuk, Alexander K. Zholkovsky. Wienna: Wiener Slawistischer Almanach, 1984. 992 c.

163. *Mikhailov D. V.* Application Of The Predicate Word's Lexical Meanings's System For Automation Of Updating Of The Dictionary Of Government Patterns / D. V. Mikhailov, G. M. Emelyanov // Interactive Systems And Technologies: The Problems of Human-Computer Interaction. Collections of scientific papers / ULSTU. Ulyanovsk, 2005. P. 164–168.

164. *Mikhailov D. V.* Clusterization of Semantic Meanings in the Problem of Sense Equivalence Situation Recognition / G. M. Emel'yanov, D. V. Mikhailov // Pattern Recognition and Image Analysis. 2009. Vol. 19, N 1. P. 92–102.

165. *Mikhailov D. V.* Filling in the Government-Pattern Dictionary in the Analysis of Equivalence for Sense Images of Statements / G. M. Emel'yanov, D. V. Mikhailov // Pattern Recognition and Image Analysis. 2007. Vol. 17, N 2. P. 268–273.

166. *Mikhailov D. V.* Formalization of the word's lexical meaning in a problem of recognition of natural language's statements's synonymy's situations / G. M.

Emelyanov, D. V. Mikhailov // 8th Int. Conf. “Pattern Recognition and Image Analysis: new Information Technologies” (PRIA-8-2007). Conf. Proc. / Mari State Technical University. Yoshkar-Ola, 2007. Vol. 2. P. 253–257.

167. *Mikhailov D. V.* Formation and clustering of noun contexts within the framework of Splintered Values / D. V. Mikhailov, G. M. Emelyanov, N. A. Stepanova // Pattern Recognition and Image Analysis. 2009. Vol. 19, N 4. P. 664–672.

168. *Mikhailov D. V.* Formation and clustering of Russian’s nouns’s contexts within the frameworks of splintered values / D. V. Mikhailov, G. M. Emelyanov // 9th Int. Conf. on Pattern Recognition and Image Analysis: New Information Technologies (PRIA-9-2008). Conf. Proc. / N.I. Lobachevsky State University of Nizhni Novgorod. Nizhni Novgorod, 2008. Vol. 2. P. 39–42.

169. *Mikhailov D. V.* Forming and clustering of syntactic relations on the bases of Natural Language’s using’s situations / D. V. Mikhailov, G. M. Emelyanov // Interactive Systems And Technologies: The Problems of Human-Computer Interaction. Collections of scientific papers / ULSTU. Ulyanovsk, 2009. Vol. III. P. 295–307.

170. *Mikhailov D. V.* Recognition of Superphrase Unities in Texts while Establishing Their Semantic Equivalence / G. M. Emelyanov, D. V. Mikhailov, E. I. Zaitseva // Pattern Recognition and Image Analysis. 2003. Vol. 13, N 3. P. 447–451.

171. *Mikhailov D. V.* Roles’s contents of word’s lexical meaning’s in a problem of recognition of synonymy’s situations on the basis of standard lexical functions / D. V. Mikhailov, G. M. Emelyanov // Interactive Systems And Technologies: The Problems of Human-Computer Interaction. Collections of scientific papers / ULSTU. Ulyanovsk, 2007. P. 159–165.

172. *Mikhailov D. V.* Semantic Clustering and Affinity Measure of Subject-Oriented Language Texts / D. V. Mikhailov, G. M. Emel’yanov // Pattern Recognition and Image Analysis. 2010. Vol. 20, N 3. P. 376–385.

173. *Mikhailov D. V.* Semantic clustering in a problem of text information’s compression / D. V. Mikhailov, G. M. Emelyanov // 10th Int. Conf. on Pattern Recog-

nition and Image Analysis: New Information Technologies (PRIA-10-2010). Conf. Proc. St. Petersburg, 2010. Vol. 2. P. 193–196.

174. *Mikhailov D. V.* Sense's Standards and Machine Understanding of Texts in the System for Computer-Aided Testing of Knowledge / G. M. Emelyanov, D. V. Mikhailov // Pattern Recognition and Image Analysis. 2011. Vol. 21, N 4. P. 705–719.

175. *Mikhailov D. V.* Sense's standards and text's understanding in computer-aided testing of knowledge / D. V. Mikhailov, G. M. Emelyanov // Interactive Systems And Technologies: The Problems of Human-Computer Interaction. Collections of scientific papers / ULSTU. Ulyanovsk, 2011. P. 318–343.

176. *Mikhailov D. V.* Synonymic Transformations in Analysis of Semantic Pattern Equivalence at the Superphrase Unity Level / G. M. Emelyanov, D. V. Mikhailov, E. I. Zaitseva // Pattern Recognition and Image Analysis. 2003. Vol. 13, N 1. P. 21–23.

177. *Mikhailov D. V.* Updating of the language knowledge base in the problem of statement's semantic images's equivalence's analysis / G. M. Emelyanov, D. V. Mikhailov // 7th Int. Conf. on Pattern Recognition and Image Analysis: New Information Technologies (PRIA-7-2004). Conf. Proc. / SPbETU. St. Petersburg, 2004. Vol. II. P. 462–465.

178. *Mikhailov D. V.* Updating the Language Knowledge Base in the Problem of Equivalence Analysis of Semantic Images of Statements / G. M. Emelyanov, D. V. Mikhailov // Pattern Recognition and Image Analysis. 2005. Vol. 15, N 2. P. 384–386.

179. *Priss Uta.* Linguistic Applications of Formal Concept Analysis / Uta Priss // Formal Concept Analysis, Foundations and Applications / Ganter; Stumme; Wille [eds.]. Berlin: Springer Verlag. LNAI 3626, 2005. P. 149–160.

180. *Shamshev A. B.* Expert Modes in the Linguistic Processor LINA / A. B. Shamshev // Interactive Systems And Technologies: The Problems of Human-Computer Interaction. Collections of scientific papers / ULSTU. Ulyanovsk, 2009. Vol. III. P. 186–191.

181. *Shamshev A. B.* The Methods of Semiotic Definiteness Assurance in Conceptual Design of Computer-Based Systems / A. B. Shamshev // Interactive Systems And Technologies: The Problems of Human-Computer Interaction. Collections of scientific papers / ULSTU. Ulyanovsk, 2009. Vol. III. P. 182–185.

182. *Sosnin P.* Question-Answer Programming in Collaborative Activity Environments / P. Sosnin // Interactive Systems And Technologies: The Problems of Human-Computer Interaction. Collections of scientific papers / ULSTU. Ulyanovsk, 2009. Vol. III. P. 10–21.

183. *Stepanova Nadezhda.* Knowledge acquisition process modeling for question answering systems / Nadezhda Stepanova, Gennady Emelyanov // Когнитивное моделирование в лингвистике: труды IX Междунар. конф. / Казанский гос. ун-т. Казань, 2007. С. 344–354.

184. The Concept Explorer [Электронный ресурс]. Режим доступа: http://conexp.sourceforge.net (дата обращения: 24.06.2011).

185. ToscanaJ: Welcome to the ToscanaJ Suite [Электронный ресурс]. Режим доступа: http://toscanaj.sourceforge.net (дата обращения: 24.06.2011).

186. *Turney Peter D.* The latent relation mapping engine: Algorithm and experiments / Peter D. Turney // Journal of Artificial Intelligence Research. 2008. N 33. P. 615–655.

187. *Zhukov S. V.* Approach to Materializing the Metrics of Information Technology Security / S. V. Zhukov, P. I. Sosnin // Interactive Systems And Technologies: The Problems of Human-Computer Interaction. Collections of scientific papers / ULSTU. Ulyanovsk, 2009. Vol. III. P. 111–115.

188. *Zhukov S. V.* Modeling Organizational Structures in Collaborative Development Environment / S. V. Zhukov // Interactive Systems And Technologies: The Problems of Human-Computer Interaction. Collections of scientific papers / ULSTU. Ulyanovsk, 2009. Vol. III. P. 116–119.

Приложение 1

Программа формирования модели ситуации языкового употребления на основе семантически эквивалентных фраз. Фрагменты исходного текста на языке Visual Prolog 5.2.

Домены пользовательских типов (файл make_se_situations.inc)

```
rlist=real*

char_list=char*

list_of_char_list=char_list*

list_of_ilist=ilist*

/* Совпадения-несовпадения буквенного состава слова для выделения флектив-
   ной части, описывается структурой word_considering:
   первый объект структуры – порядковый номер слова (для слов, нашедших
   прообразы со сходной неизменной частью);
   второй объект – совпадающая часть слова;
   третий объект – несовпадающая часть;
   четвертый объект – флаг “рассмотрено”. */

word_considering=word_considering(integer,char_list,char_list,string)
sentence_considering=word_considering*
set_of_sentences_considering=sentence_considering*
list_of_set_of_sentences_considering=set_of_sentences_considering*
```

```
/* Вспомогательные структуры для поиска прообразов с минимумом несовпаде-
   ний. */

word_considering_aux=
                word_considering_aux(integer,char_list,char_list,char_list)
word_considering_aux_list=word_considering_aux*

word_considering_aux_incoincident=
                        word_considering_aux_incoincident(integer,integer)
word_considering_aux_incoincident_list=
                                    word_considering_aux_incoincident*

/* Часть слова, не меняющаяся
   при синонимическом преобразовании. */

invariant_part=invariant_part(integer,char_list)
invariant_part_list=invariant_part*

non_invariant_parts_for_given_invariant=
        non_invariant_parts_for_given_invariant(char_list,list_of_char_list)
non_invariant_parts=non_invariant_parts_for_given_invariant*

/* Описание кластера для заданного буквенного инварианта. */

cluster_for_words_with_symbolic_invariant=
        cluster_for_words_with_symbolic_invariant(char_list,
                                                  sentence_considering)
set_of_clusters_for_words_with_symbolic_invariant=
        cluster_for_words_with_symbolic_invariant*
```

Головной модуль программы (файл make_se_situations.pro)

```
include "make_se_situations.inc"
include "make_se_situations.con"
include "hlptopic.con"

predicates
        nondeterm clustering_start(set_of_sentences_considering,
                                   invariant_part_list,
                                   set_of_sentences_considering, ilist).
        nondeterm false_taxons_reveal_with_invariants
                                        (set_of_sentences_considering,
                                        non_invariant_parts,
                                        invariant_part_list,
                                        set_of_sentences_considering,
                                        integer).
        nondeterm efapawwaraftm(set_of_sentences_considering,
                                   non_invariant_parts,
                                   set_of_sentences_considering).
        nondeterm taxons_formation_for_given_pseudophrases_set
                        (set_of_sentences_considering,non_invariant_parts).
        invariants_numbering_for_given_non_invariant_parts
                        (integer,non_invariant_parts,invariant_part_list).
        nondeterm pstnipfic(set_of_sentences_considering,
                            non_invariant_parts, invariant_part_list,
                            set_of_sentences_considering).
        nondeterm invariants_numbers_gather(invariant_part_list,ilist).
        nondeterm orders_of_words_in_sentences
                                (set_of_sentences_considering, list_of_ilist).
```

```
nondeterm most_significant_indexes_reveal(ilist,list_of_ilist,ilist).
nondeterm words_more_similar_than_differ(char_list,
                                         char_list,char_list).
nondeterm common_prefix(char_list,char_list,char_list).
nondeterm prefix(char_list,char_list,char_list).
nondeterm words_more_similar_than_differ_with_given_search
                    (word_considering,sentence_considering,
                    sentence_considering,sentence_considering,
                    list_of_char_list).
nondeterm words_in_falsetaxon_checking(list_of_char_list,
                                       char_list,char_list).
nondeterm false_taxons_reveal_in_sentence(sentence_considering,
                set_of_clusters_for_words_with_symbolic_invariant,
                sentence_considering).
nondeterm false_taxons_reveal(set_of_sentences_considering,
                set_of_clusters_for_words_with_symbolic_invariant,
                set_of_sentences_considering).
nondeterm false_taxons_merging_with_given(char_list,
                set_of_clusters_for_words_with_symbolic_invariant,
                set_of_sentences_considering,
                set_of_clusters_for_words_with_symbolic_invariant).
nondeterm false_taxons_merging
                  (set_of_clusters_for_words_with_symbolic_invariant,
                  list_of_set_of_sentences_considering).
nondeterm invariants_for_words_in_false_taxons(integer,
                                    set_of_sentences_considering,
                                    set_of_sentences_considering,
                                    invariant_part_list,integer).
```

```
nondeterm pair_of_phrases_processing(string,integer,integer,
                sentence_considering,sentence_considering,
                sentence_considering,sentence_considering,integer).
nondeterm invariant_part_list_building_for_pair(string,
                        sentence_considering,invariant_part_list).
nondeterm phrases_check_by_invariant(string,
                sentence_considering,invariant_part_list,
                invariant_part_list,set_of_sentences_considering).
nondeterm phrase_check_by_invariant(string,
                        invariant_part_list,sentence_considering,
                        sentence_considering,invariant_part_list).
nondeterm phrases_transform_invariant_respecting(string,
                    set_of_sentences_considering,invariant_part_list,
                    set_of_sentences_considering).
nondeterm non_invariant_parts_for_given_invariants
                (invariant_part_list,set_of_sentences_considering,
                non_invariant_parts).
nondeterm non_invariant_parts_for_given_invariant_search
                     (char_list,set_of_sentences_considering,
                     set_of_sentences_considering,list_of_char_list).
nondeterm nipfgisiss(char_list,sentence_considering,
                       list_of_char_list,sentence_considering).
nondeterm false_taxons_transform(integer,
                                list_of_set_of_sentences_considering,
                                set_of_sentences_considering,
                                invariant_part_list,integer).
nondeterm false_taxon_search_for_given_alphabetic_structure
                                    (char_list,non_invariant_parts,
                                    char_list,char_list).
```

```
nondeterm efpawwaraftm(sentence_considering,
                       non_invariant_parts,sentence_considering).
nondeterm taxon_transforming_respecting_new_invariant(char_list,
                       list_of_char_list,list_of_char_list).
nondeterm search_a_word_with_maximal_affinity_for_given
                       (char_list,sentence_considering,char_list,
                       char_list,char_list,char_list).
nondeterm word_and_phrase_processing(integer,char_list,
                       sentence_considering,word_considering_aux_list,
                       word_considering_aux_list,integer).
word_considering_aux_incoincident_estimate
                       (word_considering_aux_list,word_considering_aux_list,
                       word_considering_aux_incoincident_list).
potential_invariant_taxonomy_estimate(sentence_considering,rlist).
nondeterm pitcfe(sentence_considering).
nondeterm taxon_formation_for_given_invariant(char_list,char_list,
                       set_of_sentences_considering,
                       set_of_sentences_considering,
                       list_of_char_list,list_of_char_list,
                       sentence_considering,sentence_considering).
nondeterm taxon_formation_for_given_pseudophrase
                       (sentence_considering,set_of_sentences_considering,
                       set_of_sentences_considering,non_invariant_parts).
nondeterm wdtnipfic(word_considering,non_invariant_parts,
                       invariant_part_list,word_considering).
nondeterm ptnipfic(sentence_considering,non_invariant_parts,
                       invariant_part_list,sentence_considering).
nondeterm frequency_of_occurence(integer,list_of_ilist,integer).
```

```
nondeterm frequencies_of_occurence(ilist,list_of_ilist,
                                   word_considering_aux_incoincident_list).
nondeterm orders_set_for_most_significant_index(ilist,
                                                list_of_ilist,list_of_ilist).
nondeterm orders_set_for_most_significant_indexes(ilist,integer,
                                   word_considering_aux_incoincident_list,
                                   list_of_ilist,ilist).
nondeterm pair_of_phrases_processing2(sentence_considering,
                                      sentence_considering,
                                      word_considering_aux_list,
                                      word_considering_aux_list).
nondeterm pair_of_phrases_processing1(integer,
                                      word_considering_aux_list,
                                      sentence_considering,
                                      word_considering_aux_list,
                                      word_considering_aux_list).
gather_words_from_word_considering_aux
                                      (word_considering_aux_list,
                                      list_of_char_list).
nondeterm select_by_estimations
                                   (word_considering_aux_incoincident_list,
                                    word_considering_aux_list,
                                    word_considering_aux_list,
                                    word_considering_aux_list,
                                    word_considering_aux_list).
renumbering(integer,word_considering_aux_list,
                                    word_considering_aux_list,
                                    word_considering_aux_list,
                                    word_considering_aux_list,integer).
```

```
nondeterm setting_revealed_conformities(string,
                                        word_considering_aux_list,
                                        word_considering_aux_list,
                                        sentence_considering,
                                        sentence_considering,
                                        sentence_considering,
                                        sentence_considering).
nondeterm setting_revealed_conformity(string,
                                      word_considering_aux,
                                      sentence_considering,
                                      sentence_considering).
nondeterm order_of_words_in_sentence(sentence_considering,ilist).
pcbiaptnit(string,sentence_considering,invariant_part_list,
           sentence_considering,invariant_part_list,
           invariant_part_list).
phrase_check_by_new_invariant(string,invariant_part_list,
                              sentence_considering,sentence_considering).
invariant_to_pseudophrase_transform(invariant_part_list,
                                    sentence_considering).
nondeterm pseudophrase_to_new_invariant_transform(string,
                                                  invariant_part_list,
                                                  sentence_considering,
                                                  invariant_part_list).
search_pseudophrase_for_invariant(string,sentence_considering,
                                  char_list,char_list).
phrase_check_by_new_invariant_word(string,invariant_part,
                                   sentence_considering,
                                   sentence_considering).
```

```
nondeterm word_transform_invariant_respecting(string,
                                              word_considering,
                                              invariant_part_list,
                                              word_considering).
nondeterm phrase_transform_invariant_respecting(string,
                                    sentence_considering,invariant_part_list,
                                    sentence_considering).
nondeterm sort_hoar1(word_considering_aux_list,
                                    word_considering_aux_list).
nondeterm sort_hoar2(word_considering_aux_incoincident_list,
                                    word_considering_aux_incoincident_list).
nondeterm sort_hoar10(word_considering_aux_incoincident_list,
                                     word_considering_aux_incoincident_list).
nondeterm sort_hoar11(rlist,rlist).
nondeterm partition1(word_considering_aux_list,
                                    word_considering_aux,
                                    word_considering_aux_list,
                                    word_considering_aux_list).
nondeterm partition2(word_considering_aux_incoincident_list,
                                    word_considering_aux_incoincident,
                                    word_considering_aux_incoincident_list,
                                    word_considering_aux_incoincident_list).
nondeterm partition9(rlist,real,rlist,rlist).
list_len(list_of_char_list,integer).
list_len(char_list,integer).
list_len(word_considering_aux_incoincident_list,integer).
list_len(sentence_considering,integer).
list_len(set_of_sentences_considering,integer).
list_len(ilist,integer).
```

```
list_len(list_of_ilist,integer).
append(rlist,rlist,rlist).
append(word_considering_aux_list,
        word_considering_aux_list,
        word_considering_aux_list).
append(word_considering_aux_incoincident_list,
        word_considering_aux_incoincident_list,
        word_considering_aux_incoincident_list).
append(sentence_considering,
        sentence_considering,
        sentence_considering).
append(set_of_sentences_considering,
        set_of_sentences_considering,
        set_of_sentences_considering).
append(invariant_part_list,invariant_part_list,invariant_part_list).
append(char_list,char_list,char_list).
append(non_invariant_parts,
        non_invariant_parts,
        non_invariant_parts).
append(set_of_clusters_for_words_with_symbolic_invariant,
        set_of_clusters_for_words_with_symbolic_invariant,
        set_of_clusters_for_words_with_symbolic_invariant).
nondeterm append(ilist,ilist,ilist).
append(list_of_ilist,list_of_ilist,list_of_ilist).
append(non_predicates_quantity_for_sentences,
        non_predicates_quantity_for_sentences,
        non_predicates_quantity_for_sentences).
nondeterm delete(ilist,list_of_ilist,list_of_ilist).
```

```
nondeterm delete(word_considering,
                 sentence_considering,
                 sentence_considering).
nondeterm member(char,char_list).
nondeterm member(integer,ilist).
nondeterm member(word_considering_aux,
                   word_considering_aux_list).
nondeterm member(char_list,list_of_char_list).
nondeterm member(word_considering,sentence_considering).
nondeterm member(sentence_considering,
                   set_of_sentences_considering).
nondeterm member(ilist,list_of_ilist).
nondeterm member(invariant_part,invariant_part_list).
nondeterm list_set(list_of_char_list,list_of_char_list).
nondeterm list_set(list_of_ilist,list_of_ilist).
nondeterm first_n(word_considering_aux_incoincident_list,
                  integer,
                  word_considering_aux_incoincident_list,
                  word_considering_aux_incoincident_list).
nondeterm unit_sets(word_considering_aux_list,
                    word_considering_aux_list,
                    word_considering_aux_list).
nondeterm unit_sets(sentence_considering,
                    sentence_considering,
                    sentence_considering).
nondeterm unit_sets(ilist,ilist,ilist).
nondeterm unit_sets(list_of_ilist,
                    list_of_ilist,
                    list_of_ilist).
```

```
        nondeterm unit_sets(list_of_char_list,
                            list_of_char_list,
                            list_of_char_list).
        nondeterm put(integer,ilist,ilist).
        nondeterm put(char_list,list_of_char_list,list_of_char_list).
        nondeterm put(word_considering,
                      sentence_considering,
                      sentence_considering).
        nondeterm sub_set(sentence_considering,sentence_considering).
        nondeterm sub_set(ilist,ilist).
        nondeterm min(integer,integer,integer).

clauses
/* Таксономия буквенных инвариантов. Исходные данные:
   SynPhraseList_WordsLists_considering_init – список из списков структур типа
      word_considering для исходного СЭ-множества. Неизменная часть каждого
      слова представлена пустым списком.
   Результаты:
   NumberedInvariantParts – список нумерованных описаний буквенного состава
      тех частей слов, которые не меняются при синонимическом перифразирова-
      нии;
   SynPhraseListTr – список, получаемый из исходного списка
      SynPhraseList_WordsLists_considering_init путем выделения неизменяемых и
      флективных частей слов с учетом найденных буквенных инвариантов;
   IndexesForSearch – выявленное множество индексов для буквенных инвариан-
      тов с наибольшей совокупной частотой встречаемости в анализируемых ЕЯ-
      фразах. */
```

```
clustering_start(SynPhraseList_WordsLists_considering_init,
                 NumberedInvariantParts,
                 SynPhraseListTr,
                 IndexesForSearch): –
    false_taxons_reveal_with_invariants
                      (SynPhraseList_WordsLists_considering_init,
                      FalseTaxonsReprRes,
                      InvarsForFalseTaxonsRes,
                      NotInFalseTaxons1,
                      Next_Counter_of_coincidents),
    efapawwaraftm(NotInFalseTaxons1,
                  FalseTaxonsReprRes,
                  NotInFalseTaxons),
    taxons_formation_for_given_pseudophrases_set(NotInFalseTaxons,
                                                 InvariantParts),
    invariants_numbering_for_given_non_invariant_parts
                                        (Next_Counter_of_coincidents,
                                        InvariantParts,
                                        InvarsForOthers),
    append(InvarsForFalseTaxonsRes,
           InvarsForOthers,
           NumberedInvariantParts),
    append(FalseTaxonsReprRes,
          InvariantParts,
          InvariantPartsWithEndings),
    pstnipfic(SynPhraseList_WordsLists_considering_init,
           InvariantPartsWithEndings,
           NumberedInvariantParts,
           SynPhraseListTr),
```

```
    invariants_numbers_gather(NumberedInvariantParts,
                              RevealedIndexes),
    orders_of_words_in_sentences(SynPhraseListTr,IndexSequences),

    most_significant_indexes_reveal(RevealedIndexes,
                                    IndexSequences,
                                    IndexesForSearch).
```

/* Разделение ситуации СЭ – НАЧАЛО. */

/* Поиск в предложении слов, для которых буквенный состав
имеет с заданным словом больше сходств,
чем различий и которые могут образовать ложные таксоны.

Пример: “метро” (трансп.) – “метр” (ед. изм.) – НАЧАЛО. */

```
words_more_similar_than_differ(Symbols1, Symbols2,
                               Conterminous_part): –
    common_prefix(Symbols1,Symbols2,Conterminous_part),
    prefix(Conterminous_part,Symbols1,Incoincident_part1),
    prefix(Conterminous_part,Symbols2,Incoincident_part2),
    list_len(Conterminous_part,Conterminous_part_len),
    list_len(Incoincident_part1,Incoincident_part1_len),
    list_len(Incoincident_part2,Incoincident_part2_len),
    Conterminous_part_len>=Incoincident_part1_len,
    Conterminous_part_len>=Incoincident_part2_len.

words_more_similar_than_differ_with_given_search(_,[ ],[ ],[ ]).
```

```
words_more_similar_than_differ_with_given_search
            (word_considering(0,[ ],Symbols1,"false"),
            [word_considering(0,[ ],Symbols2,"false")|InitSentence],
            [word_considering(0,[ ],Symbols2,"false")|FalseTaxon],
            Others,
            [Conterminous_part|Conterminous_parts]): –
    words_more_similar_than_differ(Symbols1,Symbols2,
                                   Conterminous_part),
    words_more_similar_than_differ_with_given_search
                        (word_considering(0,[],Symbols1,"false"),
                        InitSentence, FalseTaxon, Others,
                        Conterminous_parts).

words_more_similar_than_differ_with_given_search
            (word_considering(0,[ ],Symbols1,"false"),
            [word_considering(0,[ ],Symbols2,"false")|InitSentence],
            FalseTaxon,
            [word_considering(0,[ ],Symbols2,"false")|Others],
            Conterminous_parts): –
    not(words_more_similar_than_differ(Symbols1,Symbols2,_)),
    words_more_similar_than_differ_with_given_search
                        (word_considering(0,[ ],Symbols1,"false"),
                        InitSentence,
                        FalseTaxon,
                        Others,
                        Conterminous_parts).

words_in_falsetaxon_checking([ ],Invariant,Invariant).
```

```
words_in_falsetaxon_checking([Symbols1|Conterminous_parts],
                             Symbols2, Invariant): –
    words_more_similar_than_differ(Symbols1,Symbols2,
                                   Conterminous_part),
    words_in_falsetaxon_checking(Conterminous_parts,
                                 Conterminous_part,Invariant).

false_taxons_reveal_in_sentence([ ],[ ],[ ]).

false_taxons_reveal_in_sentence([Word|Sentence],
                    [cluster_for_words_with_symbolic_invariant
                                   (Invariant,
                                   [Word|FalseTaxon])|
                    FalseTaxons],Others): –
    words_more_similar_than_differ_with_given_search(Word,
                                   Sentence, FalseTaxon,
                                   NotInFalseTaxon,
                                   Conterminous_parts),
    list_len(FalseTaxon,FalseTaxonLen),
    FalseTaxonLen>=1,
    Word=word_considering(_,_,Symbols,_),
    words_in_falsetaxon_checking(Conterminous_parts,
                                 Symbols,Invariant),
    false_taxons_reveal_in_sentence(NotInFalseTaxon,
                                   FalseTaxons,Others).

false_taxons_reveal_in_sentence([Word|Sentence],FalseTaxons,
                                [Word|Others]): –
```

```
    words_more_similar_than_differ_with_given_search(
                                                Word,Sentence,[ ],
                                                NotInFalseTaxon,[ ]),
    false_taxons_reveal_in_sentence(NotInFalseTaxon,
                                    FalseTaxons,Others).

false_taxons_reveal([ ],[ ],[ ]).

false_taxons_reveal([Sentence|Sentences],
                    FalseTaxons,
                    [NotInFalseTaxonsForSentence|NotInFalseTaxons]): –
    false_taxons_reveal_in_sentence(Sentence,
                                    FalseTaxonsForSentence,
                                    NotInFalseTaxonsForSentence),
    false_taxons_reveal(Sentences,FalseTaxons1,NotInFalseTaxons),
    append(FalseTaxonsForSentence,FalseTaxons1,FalseTaxons).

false_taxons_merging_with_given(_,[ ],[ ],[ ]).

false_taxons_merging_with_given(Invariant1,
                                [cluster_for_words_with_symbolic_invariant(
                                    Invariant2,FalseTaxon)|FalseTaxons],
                                [FalseTaxon|FalseTaxonsForGiven],
                                OthersFalseTaxons): –
    words_more_similar_than_differ(Invariant1,Invariant2,_),
    false_taxons_merging_with_given(Invariant1,FalseTaxons,
                                    FalseTaxonsForGiven,
                                    OthersFalseTaxons).
```

```
false_taxons_merging_with_given(Invariant1,
                [cluster_for_words_with_symbolic_invariant(Invariant2,
                                                FalseTaxon)|FalseTaxons],
                FalseTaxonsForGiven,
                [cluster_for_words_with_symbolic_invariant(Invariant2,
                                                    FalseTaxon)|
                                                    OthersFalseTaxons]): –
    not(words_more_similar_than_differ(Invariant1,Invariant2,_)),
    false_taxons_merging_with_given(Invariant1,
                                    FalseTaxons,
                                    FalseTaxonsForGiven,
                                    OthersFalseTaxons).

false_taxons_merging([ ],[ ]).

false_taxons_merging([cluster_for_words_with_symbolic_invariant
                                    (Invariant,FalseTaxon)|FalseTaxons],
                            [[FalseTaxon|FalseTaxonsForGiven]|Res]): –
    false_taxons_merging_with_given(Invariant,FalseTaxons,
                                    FalseTaxonsForGiven,
                                    OthersFalseTaxons),
    false_taxons_merging(OthersFalseTaxons,Res).

invariants_for_words_in_false_taxons(Curr_Counter_of_coincidents,
                                [Ph1,Ph2|Rest_of_FalseTaxonReprInit],
                                FalseTaxonReprRes,
                                InvarRes,
                                Next_Counter_of_coincidents): –
```

```
    list_len(Ph1,Ph1_LEN),
    pair_of_phrases_processing("true",
                                Curr_Counter_of_coincidents,
                                Ph1_LEN,Ph1,Ph2,
                                Ph1_for_test_new,
                                Ph2_for_test_new,
                                Next_Counter_of_coincidents),
    invariant_part_list_building_for_pair("true",
                                          Ph2_for_test_new,Invar),
    phrases_check_by_invariant("true",
                               Rest_of_FalseTaxonReprInit,
                               Invar,
                               InvarRes,
                               SynPhraseList_WordsLists_considering_new),
    phrases_transform_invariant_respecting("true",
                              [Ph1_for_test_new,
                               Ph2_for_test_new|
                               SynPhraseList_WordsLists_considering_new],
                              InvarRes,
                              FalseTaxonReprRes).

non_invariant_parts_for_given_invariants([ ],_,[ ]).

non_invariant_parts_for_given_invariants([invariant_part(_,Invariant)|
                                          Invariant_parts],
                         SentencesReprInit,
                         [non_invariant_parts_for_given_invariant(Invariant,
                               TaxonReprRes)|
                               InvariantPartsWithNonInvariants]): –
```

```
    non_invariant_parts_for_given_invariant_search(Invariant,
                                                   SentencesReprInit,
                                                   SentencesReprNext,
                                                   TaxonReprRes),
    non_invariant_parts_for_given_invariants(Invariant_parts,
                                             SentencesReprNext,
                                             InvariantPartsWithNonInvariants).

non_invariant_parts_for_given_invariant_search(_,[ ], [ ], [ ]).

non_invariant_parts_for_given_invariant_search(Invariant,
                                               [SentenceReprInit|SentencesReprInit],
                                               [SentenceReprNext|SentencesReprNext],
                                               Incoincidents): –
    nipfgisiss(Invariant,
               SentenceReprInit,
               Incoincidents1,
               SentenceReprNext),
    non_invariant_parts_for_given_invariant_search(Invariant,
                                                   SentencesReprInit,
                                                   SentencesReprNext,
                                                   Incoincidents2),
    unit_sets(Incoincidents1,Incoincidents2,Incoincidents).

/* Название “nipfgisiss” есть сокращение от “non invariant parts for given invariant
   search in single sentence” (англ.). */

nipfgisiss(_,[ ], [ ], [ ]).
```

```
nipfgisiss(Invariant,[word_considering(_,Invariant,
                                       Incoincident_part,_)|
                                  SentenceRepr],
           Incoincidents,
           SentenceReprRes): –
    nipfgisiss(Invariant,SentenceRepr,Incoincidents1,SentenceReprRes),
    put(Incoincident_part,Incoincidents1,Incoincidents).

nipfgisiss(Invariant,
          [word_considering(Label,Invariant1,Incoincident_part,Flag)|
           SentenceRepr], Incoincidents,
          [word_considering(Label,Invariant1,Incoincident_part,Flag)|
           SentenceReprRes]): –
    not(Invariant=Invariant1),
    nipfgisiss(Invariant, SentenceRepr, Incoincidents, SentenceReprRes).

false_taxons_transform(Res_Counter_of_coincidents,
                       [ ], [ ], [ ],
                       Res_Counter_of_coincidents).

false_taxons_transform(Curr_Counter_of_coincidents,
                       [FalseTaxon|FalseTaxons],
                       FalseTaxonsReprRes,
                       InvarsRes,Res_Counter_of_coincidents): –
    invariants_for_words_in_false_taxons
                            (Curr_Counter_of_coincidents,
                            FalseTaxon, FalseTaxonReprRes,
                            Invar,
                            Next_Counter_of_coincidents),
```

```
    false_taxons_transform(Next_Counter_of_coincidents,
                           FalseTaxons, FalseTaxonsReprRes1,
                           InvarsRes1, Res_Counter_of_coincidents),
    append(FalseTaxonReprRes,
           FalseTaxonsReprRes1,
           FalseTaxonsReprRes),
    append(Invar,InvarsRes1,InvarsRes).

false_taxons_reveal_with_invariants(Sentences,
                                    InvariantPartsWithNonInvariants,
                                    InvarsRes, NotInFalseTaxons,
                                    Next_Counter_of_coincidents): –
    false_taxons_reveal(Sentences,FalseTaxons,NotInFalseTaxons),
    false_taxons_merging(FalseTaxons,FalseTaxonsMerged),
    false_taxons_transform(1,FalseTaxonsMerged,
                           FalseTaxonsReprRes, InvarsRes,
                           Next_Counter_of_coincidents),
    non_invariant_parts_for_given_invariants(InvarsRes,
                                    FalseTaxonsReprRes,
                                    InvariantPartsWithNonInvariants).

/* Исключаем из всех предложений слова, которые попали в ложные таксоны –
   НАЧАЛО. */

false_taxon_search_for_given_alphabetic_structure(AllSymbols,
                    [non_invariant_parts_for_given_invariant(Invariant,
                                                TaxonReprRes)|
                     _InvariantPartsWithNonInvariants],
                    Invariant, Incoincident_part): –
```

```
    member(Incoincident_part,TaxonReprRes),
    append(Invariant,Incoincident_part,AllSymbols).

false_taxon_search_for_given_alphabetic_structure(AllSymbols,
                                    [_|InvariantPartsWithNonInvariants],
                                    Invariant,
                                    Incoincident_part): –
    false_taxon_search_for_given_alphabetic_structure(AllSymbols,
                                        InvariantPartsWithNonInvariants,
                                        Invariant,
                                        Incoincident_part).

/* Название “efpawwaraftm” есть сокращение от “exclude from phrase any words
   which are recognized as false taxons members” (англ.). */

efpawwaraftm([ ],_,[ ]).

efpawwaraftm([word_considering(_,Invariant,Incoincident_part,_)|
                SentenceRepr],
             InvariantPartsWithNonInvariants,
             SentenceReprRes): –
    append(Invariant,Incoincident_part,AllSymbols),
    false_taxon_search_for_given_alphabetic_structure(AllSymbols,
                                        InvariantPartsWithNonInvariants,
                                        _NewInvariant,
                                        _New_Incoincident_part),
    efpawwaraftm(SentenceRepr, InvariantPartsWithNonInvariants,
                 SentenceReprRes).
```

```
efpawwaraftm([word_considering(Label,
                               Invariant,Incoincident_part,Flag)|
              SentenceRepr],
              InvariantPartsWithNonInvariants,
              [word_considering(Label, Invariant,
                               Incoincident_part,Flag)|
               SentenceReprRes]): –
    append(Invariant,Incoincident_part,AllSymbols),
    not(false_taxon_search_for_given_alphabetic_structure
                               (AllSymbols,
                               InvariantPartsWithNonInvariants,
                               _,
                               _)),
    efpawwaraftm(SentenceRepr,
                 InvariantPartsWithNonInvariants,
                 SentenceReprRes).
```

/* Название “efapawwaraftm” есть сокращение от “exclude from all phrases any words which are recognized as false taxons members” (англ.). */

```
efapawwaraftm([ ],_,[ ]).

efapawwaraftm([SentenceRepr|SentencesRepr],
              InvariantPartsWithNonInvariants,
              [SentenceReprNew|SentencesReprNew]): –
    efpawwaraftm(SentenceRepr,
                 InvariantPartsWithNonInvariants,
                 SentenceReprNew),
```

```
    efapawwaraftm(SentencesRepr,
                  InvariantPartsWithNonInvariants,
                  SentencesReprNew).
```

/* Исключаем из всех предложений слова, которые попали в ложные таксоны – КОНЕЦ. */

/* Поиск в предложении слов, для которых буквенный состав имеет больше сходств, чем различий – КОНЕЦ. */

/* Преобразование таксона с учетом вновь выявленного инварианта – НАЧАЛО. */

```
taxon_transforming_respecting_new_invariant(_,[ ],[ ]).

taxon_transforming_respecting_new_invariant(Invariant_new,
                                            [Invariant|InvariantLists],
                                            Others_res): –
    prefix(Invariant_new,
           Invariant,
           Rest),
    taxon_transforming_respecting_new_invariant(Invariant_new,
                                                InvariantLists,
                                                Others_res1),
    put(Rest,Others_res1,Others_res).
```

/* Преобразование таксона с учетом вновь выявленного инварианта – КОНЕЦ. */

```
/* Поиск в предложении слова, максимально близкого заданному по буквенному
   составу. */

  search_a_word_with_maximal_affinity_for_given(Incoincident_part1,
                                                Sentence, Conterminous_part,
                                                New_Incoincident_part1,
                                                New_Incoincident_part2,
                                                Incoincident_part2): –
      word_and_phrase_processing(0, Incoincident_part1,
                                               Sentence, Aux1_unsorted,
                                               Aux2_unsorted, _),
      sort_hoar1(Aux1_unsorted,Aux1),
      sort_hoar1(Aux2_unsorted,Aux2),
      word_considering_aux_incoincident_estimate(Aux1, Aux2,
                                                          Aux3_unsorted),
      sort_hoar2(Aux3_unsorted,Aux3),
      Aux3=[word_considering_aux_incoincident(Counter,_)|_],
      member(word_considering_aux(Counter, Conterminous_part,
                                                New_Incoincident_part1,
                                                Incoincident_part1), Aux1),
      member(word_considering_aux(Counter, Conterminous_part,
                                                New_Incoincident_part2,
                                                Incoincident_part2), Aux2).

/* Оценка качества таксономии потенциальных инвариантов с вычислением
   оценки – НАЧАЛО. */

  potential_invariant_taxonomy_estimate([ ],[ ]).
```

```
potential_invariant_taxonomy_estimate
                                ([word_considering(_,Conterminous_part,
                                Incoincident_part,_)|ForEstim],
                                [Estimation|Estimations]): –
    list_len(Conterminous_part,Conterminous_part_len),
    list_len(Incoincident_part,Incoincident_part_len),
    Estimation=Conterminous_part_len/
                    (Conterminous_part_len+Incoincident_part_len),
    potential_invariant_taxonomy_estimate(ForEstim,Estimations).

pitcfe(ForEstim): –
    potential_invariant_taxonomy_estimate(ForEstim,
                                                Estimations_unsorted),
    sort_hoar11(Estimations_unsorted,Estimations),
    Estimations=[Min|_Others],
    Min>0.5.

/* Оценка качества таксономии потенциальных инвариантов с вычислением
   оценки – КОНЕЦ. */

/* Формирование таксона для заданного инварианта. */

taxon_formation_for_given_invariant(InvariantPartRes,
                                    InvariantPartRes,
                                    [ ], [ ],
                                    InvariantsListRes,
                                    InvariantsListRes,
                                    ForEstim, ForEstim).
```

```
taxon_formation_for_given_invariant(InvariantPartCurr,
                                    InvariantPartRes,
                                    [PseudoPhrase|PseudoPhraseSet],
                                    [NewPseudoPhrase|NewPseudoPhraseSet],
                                    InvariantsListCurr,
                                    InvariantsListRes,
                                    ForEstimCurr,
                                    ForEstimRes): –
    search_a_word_with_maximal_affinity_for_given
                                          (InvariantPartCurr,
                                          PseudoPhrase,
                                          InvariantPartNext,
                                          New_Incoincident_part1,
                                          New_Incoincident_part2,
                                          Incoincident_part2),
    put(Incoincident_part2,InvariantsListCurr,InvariantsListNext),
    put(word_considering(0,InvariantPartNext,
                                 New_Incoincident_part1,"false"),
        ForEstimCurr,
        ForEstimNext1),
    put(word_considering(0,InvariantPartNext,
                                 New_Incoincident_part2,"false"),
        ForEstimNext1,
        ForEstimNext),
    pitcfe(ForEstimNext),!,
    delete(word_considering(0,[ ], Incoincident_part2,"false"),
           PseudoPhrase,
           NewPseudoPhrase),
```

```
    taxon_formation_for_given_invariant(InvariantPartNext,
                                        InvariantPartRes,
                                        PseudoPhraseSet,
                                        NewPseudoPhraseSet,
                                        InvariantsListNext,
                                        InvariantsListRes,
                                        ForEstimNext,
                                        ForEstimRes).

taxon_formation_for_given_invariant(InvariantPartCurr,
                                    InvariantPartRes,
                                    [PseudoPhrase|PseudoPhraseSet],
                                    [PseudoPhrase|NewPseudoPhraseSet],
                                    InvariantsListCurr, InvariantsListRes,
                                    ForEstimCurr, ForEstimRes): –
    taxon_formation_for_given_invariant(InvariantPartCurr,
                                        InvariantPartRes,
                                        PseudoPhraseSet,
                                        NewPseudoPhraseSet,
                                        InvariantsListCurr,
                                        InvariantsListRes,
                                        ForEstimCurr,
                                        ForEstimRes).

/* Формирование таксона для заданного множества буквенных инвариантов. */

taxon_formation_for_given_pseudophrase([ ], PseudoPhraseSetRes,
                                       PseudoPhraseSetRes, [ ]).
```

```
taxon_formation_for_given_pseudophrase(
            [word_considering(0,[], Incoincident_part,"false")|
              PseudoPhrase],
            PseudoPhraseSetCurr,
            PseudoPhraseSetRes,
            [non_invariant_parts_for_given_invariant(InvariantPartRes,
                                                     TaxonReprRes)|
              InvariantParts]): –
    taxon_formation_for_given_invariant(Incoincident_part,
                                        InvariantPartRes,
                                        PseudoPhraseSetCurr,
                                        PseudoPhraseSetNext,
                                        [Incoincident_part],
                                        InvariantsListRes,
                                        [ ], _),
    list_len(InvariantsListRes,InvariantsListRes_Len),
    InvariantsListRes_Len>1,!,
    taxon_transforming_respecting_new_invariant(InvariantPartRes,
                                                InvariantsListRes,
                                                TaxonReprRes),
    taxon_formation_for_given_pseudophrase(PseudoPhrase,
                                           PseudoPhraseSetNext,
                                           PseudoPhraseSetRes,
                                           InvariantParts).

taxon_formation_for_given_pseudophrase([_|PseudoPhrase],
                                       PseudoPhraseSetCurr,
                                       PseudoPhraseSetRes,
                                       InvariantParts): –
```

```
    taxon_formation_for_given_pseudophrase(PseudoPhrase,
                                           PseudoPhraseSetCurr,
                                           PseudoPhraseSetRes,
                                           InvariantParts).

/* Формирование множества таксонов для заданного множества буквенных инвариантов. */

  taxons_formation_for_given_pseudophrases_set([ ], [ ]).

  taxons_formation_for_given_pseudophrases_set
                                  ([PseudoPhrase|PseudoPhraseSetCurr],
                                  InvariantParts): –
    not(PseudoPhrase=[]),!,
    taxon_formation_for_given_pseudophrase(PseudoPhrase,
                                           PseudoPhraseSetCurr,
                                           PseudoPhraseSetNext,
                                           InvariantParts1),
    taxons_formation_for_given_pseudophrases_set
                                           (PseudoPhraseSetNext,
                                           InvariantParts2),
    append(InvariantParts1,InvariantParts2,InvariantParts).

  taxons_formation_for_given_pseudophrases_set
                                         ([[ ]|PseudoPhraseSetCurr],
                                         InvariantParts): –
    taxons_formation_for_given_pseudophrases_set
                                           (PseudoPhraseSetCurr,
                                           InvariantParts).
```

```
/* Нумерация выявленных буквенных инвариантов для дальнейшего
   использования. */

   invariants_numbering_for_given_non_invariant_parts(_, [ ], [ ]).

   invariants_numbering_for_given_non_invariant_parts
                              (Curr_Counter_of_coincidents,
                              [non_invariant_parts_for_given_invariant
                                                    (InvariantPartRes,_)|
                               InvariantDescrs],
                              [invariant_part(Curr_Counter_of_coincidents,
                                          InvariantPartRes)|InvariantParts]): –
       Next_Counter_of_coincidents=Curr_Counter_of_coincidents+1,
       invariants_numbering_for_given_non_invariant_parts
                                          (Next_Counter_of_coincidents,
                                          InvariantDescrs, InvariantParts).

/* Построение множества номеров буквенных инвариантов для последующего
    исследования частотных характеристик. */

   invariants_numbers_gather([ ], [ ]).

   invariants_numbers_gather([invariant_part(Label,_Invariant)|
                                        InvariantParts], Res): –
       invariants_numbers_gather(InvariantParts,Res1),
       put(Label,Res1,Res).

/* Преобразования исходного множества фраз в соответствии с выявленными
    таксонами – НАЧАЛО. */
```

```
/* Название “wdtnipfic” есть сокращение от “word description transform non invari-
   ant parts for invariants considering” (англ.). */

   wdtnipfic(word_considering(0,[ ],Incoincident_part,"false"),
                [non_invariant_parts_for_given_invariant
                                                (SymbolsForInvariantPart,
                                                 PotentialEndings)|
                _InvariantPartsWithEndings],
                NumberedInvariantParts,
                word_considering(Label, SymbolsForInvariantPart,
                                        SomeEnding, "true")): –
       member(SomeEnding,PotentialEndings),
       append(SymbolsForInvariantPart,SomeEnding,Incoincident_part),
       member(invariant_part(Label,SymbolsForInvariantPart),
                 NumberedInvariantParts).

   wdtnipfic(WordReprInit, [_|InvariantPartsWithEndings],
                NumberedInvariantParts, WordReprRes): –
       wdtnipfic(WordReprInit, InvariantPartsWithEndings,
                    NumberedInvariantParts, WordReprRes).

/* Название “ptnipfic” есть сокращение от “phrase transform non invariant parts for
    invariants considering” (англ.). */

   ptnipfic([ ],_, _,[ ]).

   ptnipfic([WordReprInit|PhraseReprInit], InvariantPartsWithEndings,
                NumberedInvariantParts,
                [WordReprTr|PhraseReprTr]): –
```

```
    wdtnipfic(WordReprInit, InvariantPartsWithEndings,
                NumberedInvariantParts, WordReprTr),
    ptnipfic(PhraseReprInit,InvariantPartsWithEndings,
                NumberedInvariantParts,PhraseReprTr).

ptnipfic([WordReprInit|PhraseReprInit], InvariantPartsWithEndings,
            NumberedInvariantParts, [WordReprInit|PhraseReprTr]): –
    not(wdtnipfic(WordReprInit, InvariantPartsWithEndings,
                        NumberedInvariantParts, _)),
    ptnipfic(PhraseReprInit, InvariantPartsWithEndings,
                NumberedInvariantParts, PhraseReprTr).

/* Название “pstnipfic” есть сокращение от “phrases transform non invariant parts
    for invariants considering” (англ.). */

pstnipfic([ ],_,_,[ ]).

pstnipfic([PhraseInit|PhrasesInit], InvariantPartsWithEndings,
            NumberedInvariantParts, [PhraseTr|PhrasesTr]): –
    ptnipfic(PhraseInit,
                InvariantPartsWithEndings,
                NumberedInvariantParts, PhraseTr),
    pstnipfic(PhrasesInit,
                InvariantPartsWithEndings,
                NumberedInvariantParts, PhrasesTr).

/* Преобразования исходного множества фраз в соответствии с выявленными
    таксонами – КОНЕЦ. */
```

```
/* Вычисление частоты встречаемости элемента в списке. */

    frequency_of_occurence(_, [ ], 0).

    frequency_of_occurence(Elem,[Lst|T],Freq): –
        member(Elem,Lst),
        frequency_of_occurence(Elem,T,Freq1),
        Freq=Freq1+1.

    frequency_of_occurence(Elem,[Lst|T],Freq): –
        not(member(Elem,Lst)),
        frequency_of_occurence(Elem,T,Freq).

/* Вычисление частоты встречаемости каждого элемента из списка. */

    frequencies_of_occurence([ ], _, [ ]).

    frequencies_of_occurence([Elem|Others], Lst,
            [word_considering_aux_incoincident(Elem,Freq)|FrqsOcr]): –
        frequency_of_occurence(Elem,Lst,Freq),
        frequencies_of_occurence(Others,Lst,FrqsOcr).

/* Отобрать списки, включающие заданные элементы. */

    orders_set_for_most_significant_index(_, [ ], [ ]).

    orders_set_for_most_significant_index(GivenIndexes,
                                [IndexList|ListOfIndexLists],
                                [IndexList|IndexListsForGivenIndexPresense]): –
```

```
    sub_set(GivenIndexes,IndexList),
    orders_set_for_most_significant_index(GivenIndexes,
                                          ListOfIndexLists,
                                          IndexListsForGivenIndexPresense).

orders_set_for_most_significant_index(GivenIndexes,
                                      [IndexList|ListOfIndexLists],
                                      IndexListsForGivenIndexPresense): –
    not(sub_set(GivenIndexes,IndexList)),
    orders_set_for_most_significant_index(GivenIndexes,
                                          ListOfIndexLists,
                                          IndexListsForGivenIndexPresense).

orders_set_for_most_significant_indexes(IndexesForSearch,
                                        _, [ ], _,
                                        IndexesForSearch).

orders_set_for_most_significant_indexes(GivenIndexesPrev, Estimation,
        [word_considering_aux_incoincident(MostSignificantIndex,
                                           _)|_],
        ListOfIndexListsPrev, IndexesForSearch): –
    orders_set_for_most_significant_index
                                ([MostSignificantIndex|GivenIndexesPrev],
                                ListOfIndexListsPrev,
                                ListOfIndexListsNext),
    list_len([MostSignificantIndex|GivenIndexesPrev],
             Significant_indexes_number),
    list_len(ListOfIndexListsNext,Number_of_Phrases),
    EstimationNext=Significant_indexes_number*Number_of_Phrases,
```

```
    EstimationNext<Estimation,
    IndexesForSearch=GivenIndexesPrev.

orders_set_for_most_significant_indexes(GivenIndexesPrev, Estimation,
                [word_considering_aux_incoincident(MostSignificantIndex,
                                                   _)|FrqsOcr],
                ListOfIndexListsPrev, IndexesForSearch): –
    orders_set_for_most_significant_index
                                ([MostSignificantIndex|GivenIndexesPrev],
                                ListOfIndexListsPrev,
                                ListOfIndexListsNext),
    list_len([MostSignificantIndex|GivenIndexesPrev],
                Significant_indexes_number),
    list_len(ListOfIndexListsNext,Number_of_Phrases),
    EstimationNext=Significant_indexes_number*Number_of_Phrases,
    EstimationNext>=Estimation,
    orders_set_for_most_significant_indexes
                                ([MostSignificantIndex|GivenIndexesPrev],
                                EstimationNext, FrqsOcr,
                                ListOfIndexListsNext, IndexesForSearch).

/* Выделение подмножества индексов с наибольшей совокупной частотой встре-
    чаемости. */

most_significant_indexes_reveal(Indexes,
                                OrdersSet,
                                IndexesForSearch): –
    frequencies_of_occurence(Indexes, OrdersSet, FrqsOcrUnsorted),
    sort_hoar10(FrqsOcrUnsorted,FrqsOcr),
```

```
    orders_set_for_most_significant_indexes([ ], 0,
                                                    FrqsOcr,
                                                    OrdersSet,
                                                    IndexesForSearch).

/* Разделение ситуации СЭ – КОНЕЦ. */

/* Сравнение пары фраз – НАЧАЛО. */

  word_and_phrase_processing(Counter,_,[ ],[ ],[ ],Counter).

  word_and_phrase_processing(Counter, Incoincident_part1,
                [word_considering(0,[ ],Incoincident_part2,"false")|
                  SentenceRepr],
                [word_considering_aux(Counter,
                                      Conterminous_part,
                                      New_Incoincident_part1,
                                      Incoincident_part1)|Aux1],
                [word_considering_aux(Counter,
                                      Conterminous_part,
                                      New_Incoincident_part2,
                                      Incoincident_part2)|Aux2],
                CounterRes): –
    common_prefix(Incoincident_part1,
                  Incoincident_part2,
                  Conterminous_part),
    prefix(Conterminous_part,
           Incoincident_part1,
           New_Incoincident_part1),
```

```
    prefix(Conterminous_part,
           Incoincident_part2,
           New_Incoincident_part2),
    not(Conterminous_part=[ ]),
    Counter1=Counter+1,
    word_and_phrase_processing(Counter1,
                               Incoincident_part1,
                               SentenceRepr,
                               Aux1,
                               Aux2,
                               CounterRes).

word_and_phrase_processing(Counter, Incoincident_part1,
                           [_|SentenceRepr],
                           Aux1, Aux2,
                           CounterRes): –
    word_and_phrase_processing(Counter,
                               Incoincident_part1,
                               SentenceRepr,
                               Aux1,
                               Aux2,
                               CounterRes).

pair_of_phrases_processing2([word_considering(0,[ ],
                                              Incoincident_part1, "false")|
                             Sentence1Repr],
                            Sentence2_curr_considering,
                            Aux1,
                            Aux2): –
```

```
    word_and_phrase_processing(0, Incoincident_part1,
                                  Sentence2_curr_considering,
                                  Aux11, Aux22, CounterNext),
    pair_of_phrases_processing1(CounterNext,Aux22,
                                [word_considering(0, [ ],
                                                  Incoincident_part1,
                                                  "false")|
                                 Sentence1Repr],
                                Aux222, Aux111),
    unit_sets(Aux11,Aux111,Aux1),
    unit_sets(Aux22,Aux222,Aux2).

pair_of_phrases_processing1(_,[ ],_,[ ],[ ]).

pair_of_phrases_processing1(Counter,
                            [word_considering_aux(_,
                                                  _Conterminous_part,
                                                  _New_Incoincident_part2,
                                                  Incoincident_part2)|Aux2],
                            SentenceRepr,
                            Aux11,
                            Aux22): –
    word_and_phrase_processing(Counter, Incoincident_part2,
                                  SentenceRepr,
                                  Aux111, Aux222,
                                  CounterNext),
    pair_of_phrases_processing1(CounterNext, Aux2,
                                  SentenceRepr,
                                  Aux1111, Aux2222),
```

```
    unit_sets(Aux111,
              Aux1111,
              Aux11),
    unit_sets(Aux222,
              Aux2222,
              Aux22).

gather_words_from_word_considering_aux([ ], [ ]).

gather_words_from_word_considering_aux
              ([word_considering_aux(_,_,_,Word_char_list)|AuList],
              [Word_char_list|Word_char_lists]): –
    gather_words_from_word_considering_aux(AuList,
                                           Word_char_lists).

word_considering_aux_incoincident_estimate([ ], [ ], [ ]).

word_considering_aux_incoincident_estimate
        ([word_considering_aux(Counter, Conterminous_part,
                               New_Incoincident_part1, _)|Aux1],
        [word_considering_aux(Counter, Conterminous_part,
                               New_Incoincident_part2, _)|Aux2],
        [word_considering_aux_incoincident(Counter,Diff)|Aux3]): –
    list_len(New_Incoincident_part1,New_Incoincident_part1_len),
    list_len(New_Incoincident_part2,New_Incoincident_part2_len),
    Diff=New_Incoincident_part1_len+New_Incoincident_part2_len,
    word_considering_aux_incoincident_estimate(Aux1,Aux2,Aux3).

select_by_estimations([ ], _, _, [ ], [ ]).
```

```
select_by_estimations
                ([word_considering_aux_incoincident(Counter,_)|Tail],
                Aux1, Aux2,
                [word_considering_aux(Counter, Conterminous_part,
                                      New_Incoincident_part1,
                                      Wrd1Chars)|NewAux1],
                [word_considering_aux(Counter, Conterminous_part,
                                      New_Incoincident_part2,
                                      Wrd2Chars)|NewAux2]): –
    member(word_considering_aux(Counter, Conterminous_part,
                                New_Incoincident_part1,
                                Wrd1Chars), Aux1),
    member(word_considering_aux(Counter,Conterminous_part,
                                New_Incoincident_part2,
                                Wrd2Chars), Aux2),
    select_by_estimations(Tail,Aux1,Aux2,NewAux1,NewAux2).

renumbering(MaxNumber,[ ] ,[ ] ,[ ], [ ] ,MaxNumber).

renumbering(NewNumber,
                [word_considering_aux(Counter,
                                      Conterminous_part,
                                      New_Incoincident_part1,
                                      Wrd1Chars)|Aux1],
                [word_considering_aux(Counter,
                                      Conterminous_part,
                                      New_Incoincident_part2,
                                      Wrd2Chars)|Aux2],
                [word_considering_aux(NewNumber,
```

```
                                    Conterminous_part,
                                    New_Incoincident_part1,
                                    Wrd1Chars)|NewAux1],
                [word_considering_aux(NewNumber,
                                    Conterminous_part,
                                    New_Incoincident_part2,
                                    Wrd2Chars)|NewAux2],
                MaxNumber): –
    NewNumber1=NewNumber+1,
    renumbering(NewNumber1,
                Aux1, Aux2, NewAux1,
                NewAux2, MaxNumber).

setting_revealed_conformities(_, [ ], [ ],
                                Sentence1ReprRes,
                                Sentence2ReprRes,
                                Sentence1ReprRes,
                                Sentence2ReprRes).

setting_revealed_conformities(Flag,
                                [Aux1Head|Aux1Tail],
                                [Aux2Head|Aux2Tail],
                                Sentence1ReprOld, Sentence2ReprOld,
                                Sentence1ReprRes, Sentence2ReprRes): –
    setting_revealed_conformity(Flag,
                                Aux1Head,
                                Sentence1ReprOld,
                                Sentence1ReprNew),
```

```
    setting_revealed_conformity(Flag,Aux2Head,Sentence2ReprOld,
                                Sentence2ReprNew),
    setting_revealed_conformities(Flag,
                                  Aux1Tail, Aux2Tail,
                                  Sentence1ReprNew,
                                  Sentence2ReprNew,
                                  Sentence1ReprRes,
                                  Sentence2ReprRes).

setting_revealed_conformity(Flag,
                            word_considering_aux(NewLabel,
                                                 Conterminous_part,
                                                 New_Incoincident_part,
                                                 WrdChars),
                            [word_considering(0,[],WrdChars,"false")|
                              SentenceRepr],
                            [word_considering(NewLabel,
                                                 Conterminous_part,
                                                 New_Incoincident_part,
                                                 Flag)|
                              SentenceRepr]): –!.

setting_revealed_conformity(Flag,
                            word_considering_aux(NewLabel,
                                                 Conterminous_part,
                                                 New_Incoincident_part,
                                                 WrdChars),
                            [H|SentenceRepr],
                            [H|SentenceReprNew]): –
```

```
    setting_revealed_conformity(Flag,
                                word_considering_aux(NewLabel,
                                        Conterminous_part,
                                        New_Incoincident_part,
                                        WrdChars),
                                SentenceRepr,
                                SentenceReprNew).

pair_of_phrases_processing(_,
                                New_Counter_of_coincidents,
                                0, Ph1, Ph2, Ph1, Ph2,
                                New_Counter_of_coincidents).


pair_of_phrases_processing(Flag,
                                Counter_of_coincidents,
                                Words_must_be_considered,
                                Ph1_for_test,
                                Ph2_for_test,
                                Ph1_res,
                                Ph2_res,
                                Res_Counter_of_coincidents): –
    Words_must_be_considered>0,
    pair_of_phrases_processing2(Ph1_for_test,
                                Ph2_for_test,
                                Aux1_unsorted,
                                Aux2_unsorted),
    sort_hoar1(Aux1_unsorted,Aux1),
    sort_hoar1(Aux2_unsorted,Aux2),
```

```
word_considering_aux_incoincident_estimate(Aux1,
                                           Aux2,
                                           Aux3_unsorted),
sort_hoar2(Aux3_unsorted,Aux3),
gather_words_from_word_considering_aux(Aux1,WrdsAux1),
list_set(WrdsAux1,WrdsSetAux1),
gather_words_from_word_considering_aux(Aux2,WrdsAux2),
list_set(WrdsAux2,WrdsSetAux2),
list_len(WrdsSetAux1,WrdsSetLenAux1),
list_len(WrdsSetAux2,WrdsSetLenAux2),
min(WrdsSetLenAux1,WrdsSetLenAux2,WrdsSetMinLen),
first_n(Aux3,WrdsSetMinLen,Mins_from_Aux3,_),
select_by_estimations(Mins_from_Aux3,
                      Aux1,
                      Aux2,
                      NewAux1,
                      NewAux2),
sort_hoar1(NewAux1,X),
sort_hoar1(NewAux2,Y),
renumbering(Counter_of_coincidents,
            X,Y,
            X1,Y1,
            New_Counter_of_coincidents),
setting_revealed_conformities(Flag,X1,Y1,
                              Ph1_for_test,Ph2_for_test,
                              Ph1_new,Ph2_new),
Ph1_new=[Head_Ph1_new1|Tail_Ph1_new1],
append(Tail_Ph1_new1,[Head_Ph1_new1],Ph1_for_test_new),
Words_must_be_considered1=Words_must_be_considered−1,
```

```
    pair_of_phrases_processing(Flag,
                               New_Counter_of_coincidents,
                               Words_must_be_considered1,
                               Ph1_for_test_new, Ph2_new,
                               Ph1_res, Ph2_res,
                               Res_Counter_of_coincidents).

pair_of_phrases_processing(Flag,
                           Counter_of_coincidents,
                           Words_must_be_considered,
                           Ph1_for_test, Ph2_for_test,
                           Ph1_res, Ph2_res,
                           Res_Counter_of_coincidents): –
    Words_must_be_considered>0,
    not(pair_of_phrases_processing2(Ph1_for_test,Ph2_for_test,_,_)),
    Ph1_for_test=[Head_Ph1_for_test|Tail_Ph1_for_test],
    append(Tail_Ph1_for_test,[Head_Ph1_for_test],Ph1_for_test_new),
    Words_must_be_considered1=Words_must_be_considered – 1,
    pair_of_phrases_processing(Flag,
                               Counter_of_coincidents,
                               Words_must_be_considered1,
                               Ph1_for_test_new,
                               Ph2_for_test,
                               Ph1_res,
                               Ph2_res,
                               Res_Counter_of_coincidents).

/* Сравнение пары фраз – КОНЕЦ. */
```

```
/* Сбор информации о порядках слов относительно выявленного инварианта по
   предложениям. */

orders_of_words_in_sentences([ ], [ ]).

orders_of_words_in_sentences([Sentence|Sentences],
                             [Order|Orders]): –
    order_of_words_in_sentence(Sentence,Order),
    orders_of_words_in_sentences(Sentences,Orders).

order_of_words_in_sentence([ ], [ ]).

order_of_words_in_sentence([word_considering(Label,
                                             _Conterminous_part,
                                             _Incoincident_part,
                                             Flag)|RestOfPhrase],
                           [Label|LabelList]): –
    Flag="true",
    order_of_words_in_sentence(RestOfPhrase,LabelList).

order_of_words_in_sentence([word_considering(_Label,
                                             _Conterminous_part,
                                             _Incoincident_part,
                                             Flag)|RestOfPhrase],
                           LabelList): –
    not(Flag="true"),
    order_of_words_in_sentence(RestOfPhrase,LabelList).
```

```
/* Построение списка частей слова, не меняющихся при взаимном синонимическом преобразовании фраз внутри пары.
   Далее при обработке перифраз список инвариантов не пополняется, с целью выявления устойчивых сочетаний слов из него могут удаляться инварианты, не нашедшие прообразов в новых перифразах. */

   invariant_part_list_building_for_pair(_,[ ],[ ]).

   invariant_part_list_building_for_pair(Flag,
                                         [word_considering(0,[],_,"false")|SentenceRepr],
                                         Invariant_parts_list): –
       invariant_part_list_building_for_pair(Flag, SentenceRepr,
                                             Invariant_parts_list).

   invariant_part_list_building_for_pair(Flag,
                                       [word_considering(NewLabel,
                                                         Conterminous_part,
                                                         _,
                                                         Flag)|SentenceRepr],
                                       [invariant_part(NewLabel,
                                                       Conterminous_part)|
                                                       Invariant_parts_list]): –
       invariant_part_list_building_for_pair(Flag, SentenceRepr,
                                             Invariant_parts_list).

/* Проверка множества фраз с уточнением инварианта. */

   phrases_check_by_invariant(_,[ ],Invariant,Invariant,[ ]).
```

```
phrases_check_by_invariant(Flag,[Phrase|PhrasesSet],
                           CurrInvar,ResInvar,
                           [PhraseReprRes|PhrasesSetReprRes]): –
    phrase_check_by_invariant(Flag, CurrInvar, Phrase,
                              PhraseReprRes,InvarForNext),
    phrases_check_by_invariant(Flag,PhrasesSet,InvarForNext,
                              ResInvar,PhrasesSetReprRes).

/* Проверка очередной фразы по выявленному инварианту. */
/* Название “pcbiaptnit” есть сокращение от “phrase check by invariant and
   pseudophrase to new invariant transform” (англ.). */

pcbiaptnit("local", Sentence2_curr_considering, Invariant_parts_list,
           Sentence2_res_considering, Invariant_parts_list_new,
           Invariant_parts_list_res): –
    Invariant_parts_list_new=[ ],!,
    Sentence2_res_considering=Sentence2_curr_considering,
    Invariant_parts_list_res=Invariant_parts_list.

pcbiaptnit(Flag, Sentence2_curr_considering, _Invariant_parts_list,
           Sentence2_res_considering, Invariant_parts_list_new,
           Invariant_parts_list_res): –
    phrase_check_by_new_invariant(Flag,
                                  Invariant_parts_list_new,
                                  Sentence2_curr_considering,
                                  Sentence2_res_considering),
    Invariant_parts_list_res=Invariant_parts_list_new.

phrase_check_by_invariant(Flag, Invariant_parts_list,
```

```
                                Sentence2_curr_considering,
                                Sentence2_res_considering,
                                Invariant_parts_list_res): –
    invariant_to_pseudophrase_transform(Invariant_parts_list,
                                        Pseudo_Phrase),
    list_len(Pseudo_Phrase,Pseudo_Phrase_LEN),
    pair_of_phrases_processing(Flag,1, Pseudo_Phrase_LEN,
                Pseudo_Phrase,Sentence2_curr_considering,
                Pseudo_Phrase_new,Sentence2_next_considering,_),
    pseudophrase_to_new_invariant_transform(Flag,
                                        Invariant_parts_list,
                                        Pseudo_Phrase_new,
                                        Invariant_parts_list_new),
    pcbiaptnit(Flag, Sentence2_next_considering, Invariant_parts_list,
            Sentence2_res_considering, Invariant_parts_list_new,
            Invariant_parts_list_res).

/* Генерация псевдофразы для списка инвариантов. */

  invariant_to_pseudophrase_transform([ ], [ ]).

  invariant_to_pseudophrase_transform(
                [invariant_part(_Label,Conterminous_part)|
                                        Invariant_parts_list],
                [word_considering(0,[],Conterminous_part,"false")|
                                        SentenceRepr]): –
    invariant_to_pseudophrase_transform(Invariant_parts_list,
                                        SentenceRepr).
```

```
/* Новый инвариант и расстановка композиционных меток. */

  search_pseudophrase_for_invariant(Flag,
                    [word_considering(_,New_Conterminous_part,
                                      New_Incoincident_part,Flag)|
                     _SentenceReprRest],
                    Conterminous_part, New_Conterminous_part): –
      append(New_Conterminous_part,
             New_Incoincident_part,
             Conterminous_part),!.

  search_pseudophrase_for_invariant(Flag, [_|SentenceReprRest],
                                    Conterminous_part,
                                    New_Conterminous_part): –
      search_pseudophrase_for_invariant(Flag,SentenceReprRest,
                                        Conterminous_part,
                                        New_Conterminous_part).

  pseudophrase_to_new_invariant_transform(_, [ ], _, [ ]).

  pseudophrase_to_new_invariant_transform(Flag,
                    [invariant_part(Label,Conterminous_part)|
                                    Invariant_parts_list_old],
                    SentenceRepr,
                    [invariant_part(Label,New_Conterminous_part)|
                                    Invariant_parts_list_new]): –
      search_pseudophrase_for_invariant(Flag,SentenceRepr,
                                        Conterminous_part,
                                        New_Conterminous_part),
```

```
    pseudophrase_to_new_invariant_transform(Flag,
                                    Invariant_parts_list_old, SentenceRepr,
                                    Invariant_parts_list_new).

pseudophrase_to_new_invariant_transform(Flag,
                                    [invariant_part(_,Conterminous_part)|
                                                Invariant_parts_list_old],
                                    SentenceRepr,
                                    Invariant_parts_list_new): –
    not(search_pseudophrase_for_invariant(Flag, SentenceRepr,
                                                Conterminous_part,_)),
    pseudophrase_to_new_invariant_transform(Flag,
                                                Invariant_parts_list_old,
                                                SentenceRepr,
                                                Invariant_parts_list_new).

phrase_check_by_new_invariant_word(Flag,
                                invariant_part(Label,Conterminous_part),
                                [word_considering(_,Conterminous_part,
                                                Incoincident_part,Flag)|
                                  SentenceReprRest],
                                [word_considering(Label,Conterminous_part,
                                                Incoincident_part,Flag)|
                                  SentenceReprRest]): – !.

phrase_check_by_new_invariant_word(Flag,
                                invariant_part(Label,Conterminous_part),
                                [Wrd_Consider|SentenceReprRest],
                                [Wrd_Consider|SentenceReprRestNew]): –
```

```
    phrase_check_by_new_invariant_word(Flag,
                                       invariant_part(Label,Conterminous_part),
                                       SentenceReprRest,
                                       SentenceReprRestNew).

phrase_check_by_new_invariant(_,[ ], SentReprRes,SentReprRes).

phrase_check_by_new_invariant(Flag, [Invariant|Invariants],
                                    SentReprCurr, SentReprRes): –
    phrase_check_by_new_invariant_word(Flag, Invariant,
                                             SentReprCurr,
                                             SentReprNext),
    phrase_check_by_new_invariant(Flag, Invariants,
                                        SentReprNext,
                                        SentReprRes).
```

/* Окончательное преобразование СЭ-множества с учетом выявленного инварианта. */

```
word_transform_invariant_respecting(Flag,
                    word_considering(_,Conterminous_part,
                                       Incoincident_part,_),
                    [ ],
                    word_considering(0, [ ],
                                       Incoincident_part_new,
                                       "false")): –
    not(Flag="false"),
    append(Conterminous_part, Incoincident_part,
           Incoincident_part_new),!.
```

```
word_transform_invariant_respecting(Flag,
                                    word_considering(Label,
                                                     Conterminous_part,
                                                     Incoincident_part,
                                                     Flag),
                                    [invariant_part(Label,
                                                    Conterminous_part)|
                                     _Invariant_parts],
                                    word_considering(Label,
                                                     Conterminous_part,
                                                     Incoincident_part,
                                                     Flag)): – !.

word_transform_invariant_respecting(Flag,
                                    word_considering(Label,
                                                     Conterminous_part,
                                                     Incoincident_part,
                                                     Flag),
                                    [invariant_part(Label,
                                                    Conterminous_part_new)|
                                    _Invariant_parts],
                                    word_considering(Label,
                                                     Conterminous_part_new,
                                                     Incoincident_part_new,
                                                     Flag)): –
    append(Conterminous_part_new, Incoincident_part_for_add,
           Conterminous_part),!,
    append(Incoincident_part_for_add, Incoincident_part,
           Incoincident_part_new).
```

```
word_transform_invariant_respecting(Flag,
                                    word_considering(Label,
                                                     Conterminous_part,
                                                     Incoincident_part,
                                                     Flag),
                                    [invariant_part(Label1,_)|
                                     Invariant_parts],
                                    word_considering(Label_Res,
                                                     Conterminous_part_res,
                                                     Incoincident_part_res,
                                                     Flag_res)): –
    not(Label=Label1),
    word_transform_invariant_respecting(Flag,
                                        word_considering(Label,
                                                         Conterminous_part,
                                                         Incoincident_part,
                                                         Flag),
                                        Invariant_parts,
                                        word_considering(Label_Res,
                                                         Conterminous_part_res,
                                                         Incoincident_part_res,
                                        Flag_res)).

word_transform_invariant_respecting(Flag,
                                    word_considering(Label,
                                                     Conterminous_part,
                                                     Incoincident_part,
                                                     "false"),
                                    _,
```

```
                                word_considering(Label,
                                                 Conterminous_part,
                                                 Incoincident_part,
                                                 "false")): –
    not(Flag="false").

phrase_transform_invariant_respecting(_, [ ], _, [ ]).

phrase_transform_invariant_respecting(Flag,
                                      [WordRepr|PhraseRest],
                                      Invar,
                                      [WordReprNew|PhraseRestNew]): –
    word_transform_invariant_respecting(Flag,
                                        WordRepr,Invar,
                                        WordReprNew),
    phrase_transform_invariant_respecting(Flag,
                                          PhraseRest,Invar,
                                          PhraseRestNew).

phrases_transform_invariant_respecting(_, [ ], _, [ ]).

phrases_transform_invariant_respecting(Flag,
                                       [Phrase|PhrasesSet],
                                       Invar,
                                       [NewPhrase|NewPhrasesSet]): –
    phrase_transform_invariant_respecting(Flag,
                                          Phrase,
                                          Invar,
                                          NewPhrase),
```

```
        phrases_transform_invariant_respecting(Flag, PhrasesSet,
                                                Invar, NewPhrasesSet).

/* Реализация вспомогательных процедур. */
/* Является ли один список префиксом другого. */

    prefix([ ], Suffix, Suffix).

    prefix([H|T],[H|T1],Suffix): – prefix(T,T1,Suffix).

/* Имеют ли два списка общий префикс. */

    common_prefix([ ],_,[ ]).

    common_prefix(_,[ ],[ ]).

    common_prefix([H1|_],[H2|_],[ ]): – not(H1=H2).

    common_prefix([H|T1],[H|T2],[H|Res]): – common_prefix(T1,T2,Res).

/* Нахождение длины списка. */

    list_len([ ],0).
    list_len([_|Tail],Len): – list_len(Tail,Len1), Len=Len1+1.

/* Объединение двух списков. */

    append([ ],L,L).
    append([Head|Tail],Lst2,[Head|Tail_res]): – append(Tail,Lst2,Tail_res).
```

```
/* Принадлежность элемента списку. */

    member(Head,[Head|_]).
    member(Elem,[_|T]): – member(Elem,T).

/* Удаление всех вхождений заданного элемента в список. */

    delete(_,[ ],[ ]).
    delete(H,[H|T],Res): – delete(H,T,Res).
    delete(Elem,[H|T],[H|Res]): – not(H=Elem), delete(Elem,T,Res).

/* Преобразование списка в множество. */

    list_set([ ],[ ]).

    list_set([Head_lst|Tail_lst],Res): –
        member(Head_lst,Tail_lst),!, list_set(Tail_lst,Res).

    list_set([Head_lst|Tail_lst],[Head_lst|Res]): –
        list_set(Tail_lst,Res).

/* Выделение заданного числа первых элементов списка. */

    first_n(Lst,N,Lst,[ ]): – list_len(Lst,ListLen), N>=ListLen,!.

    first_n(Lst,0,[ ],Lst).

    first_n([Lst_Head|Lst_Tail],N,[Lst_Head|Rest_of_First],Rest_of_List): –
        N>0, N1=N–1, first_n(Lst_Tail,N1,Rest_of_First,Rest_of_List).
```

```
/* Объединение множеств. */

    unit_sets([ ],Set2,Set2).
    unit_sets([H|T],Set2,[H|Res]): –
          not(member(H,Set2)),
          unit_sets(T,Set2,Res).

    unit_sets([H|T],Set2,Res): –
          member(H,Set2),
          unit_sets(T,Set2,Res).

/* Правило помещает объект в список, если этот объект там отсутствует. */

    put(Obj,Arg,[Obj|Arg]): – not(member(Obj,Arg)).

    put(Obj,Arg,Arg): – member(Obj,Arg).

/* Проверка, является ли одно множество подмножеством другого. */

    sub_set([ ],_).
    sub_set([H|T],Set2):  – member(H,Set2), sub_set(T,Set2).

/* Минимум из двух чисел. */

    min(X,Y,X): – X<Y.
    min(X,Y,Y): – X>=Y.
```

```
/* Сортировка Хаара. */

   sort_hoar1([ ],[ ]).

   sort_hoar1([word_considering_aux(Counter,
                                    Common,Incoinc,
                                    WordTotal)|Tail],
              Res): –
      partition1(Tail,
                 word_considering_aux(Counter,
                                      Common,Incoinc,WordTotal),
                 Littles,
                 Bigs),
      sort_hoar1(Littles,Ls),
      sort_hoar1(Bigs,Bs),
      append(Ls,
             [word_considering_aux(Counter,
                                   Common,Incoinc,WordTotal)|Bs],
             Res).

   sort_hoar2([ ],[ ]).

   sort_hoar2([word_considering_aux_incoincident(Counter,
                                                 IncoincNum)|Tail],Res): –
      partition2(Tail,
                 word_considering_aux_incoincident(Counter,
                                                   IncoincNum),
                 Littles,
                 Bigs),
```

```
    sort_hoar2(Littles,Ls),
    sort_hoar2(Bigs,Bs),
    append(Ls,
           [word_considering_aux_incoincident(Counter,
                                              IncoincNum)|Bs],
           Res).

sort_hoar10([ ],[ ]).

sort_hoar10([word_considering_aux_incoincident(Counter,
                                               IncoincNum)|Tail],
            Res): –
    partition2(Tail,
               word_considering_aux_incoincident(Counter,
                                                 IncoincNum),
               Littles,
               Bigs),
    sort_hoar10(Littles,Ls),
    sort_hoar10(Bigs,Bs),
    append(Bs,
           [word_considering_aux_incoincident(Counter,
                                              IncoincNum)|Ls],
           Res).

sort_hoar11([ ],[ ]).

sort_hoar11([Head|Tail],Res): –
    partition9(Tail,Head,Littles,Bigs),
    sort_hoar11(Littles,Ls),
```

```
    sort_hoar11(Bigs,Bs),
    append(Ls,[Head|Bs],Res).

/* Разделение списка на “большие” и “меньшие” относительно заданного барье-
    ра. */

partition1([ ],_,[ ],[ ]).

partition1([word_considering_aux(Counter,
                                 Common,Incoinc,
                                 WordTotal)|Tail],
           word_considering_aux(Barier,BCommon,
                                 BIncoinc,BWordTotal),
           [word_considering_aux(Counter,Common,
                                 Incoinc,WordTotal)|Littles],
           Bigs): –
    Counter<=Barier,
    partition1(Tail,
               word_considering_aux(Barier,BCommon,
                                    BIncoinc,BWordTotal),
               Littles, Bigs).

partition1([word_considering_aux(Counter,
                                 Common,Incoinc,
                                 WordTotal)|Tail],
           word_considering_aux(Barier,
                                 BCommon,
                                 BIncoinc,
                                 BWordTotal),
```

```
            Littles,
            [word_considering_aux(Counter,
                                  Common,
                                  Incoinc,
                                  WordTotal)|Bigs]): –
    Counter>Barier,
    partition1(Tail,
            word_considering_aux(Barier,
                                 BCommon,
                                 BIncoinc,
                                 BWordTotal),
            Littles,
            Bigs).

partition2([ ],_,[ ],[ ]).

partition2([word_considering_aux_incoincident(Counter,
                                              Incoinc)|Tail],
            word_considering_aux_incoincident(BCounter,BIncoinc),
            [word_considering_aux_incoincident(Counter,
                                              Incoinc)|Littles],
            Bigs): –
    Incoinc<=BIncoinc,
    partition2(Tail,
            word_considering_aux_incoincident(BCounter,
                                              BIncoinc),
            Littles,
            Bigs).
```

```
partition2([word_considering_aux_incoincident(Counter,Incoinc)|Tail],
            word_considering_aux_incoincident(Barier,BIncoinc),
            Littles,
            [word_considering_aux_incoincident(Counter,Incoinc)|Bigs]): –
        Incoinc>BIncoinc,
        partition2(Tail,
                    word_considering_aux_incoincident(Barier,BIncoinc),
                    Littles,
                    Bigs).

partition9([ ],_,[ ],[ ]).

partition9([Head|Tail], Barier, [Head|Littles], Bigs): –
        Head<=Barier, partition9(Tail,Barier,Littles,Bigs).

partition9([Head|Tail], Barier, Littles, [Head|Bigs]): –
        Head>Barier, partition9(Tail,Barier,Littles,Bigs).
```

Приложение 2

АКТЫ ОБ АПРОБАЦИИ РЕЗУЛЬТАТОВ ДИССЕРТАЦИОННОЙ РАБОТЫ.

УТВЕРЖДАЮ
Проректор по НР НовГУ
_____ Бондаренко Е.А.
“ ” _____ 2012 г.

УТВЕРЖДАЮ
Генеральный директор
ОАО “НИИПТ “Растр”
_____ Челпанов В.И.
“ ” _____ 2012 г.

А К Т

о результатах опытной эксплуатации

Мы, нижеподписавшиеся, представители НовГУ в лице зав. кафедрой ИТиС Гаврикова А.Л. и научного руководителя ГБ НИР № 0120.0 704719 Емельянова Г.М. (с одной стороны) и представители ОАО “НИИПТ “Растр” в лице ведущего научного сотрудника, к.т.н. Смирнова Н.И. (с другой стороны) составили настоящий акт о нижеследующем:

1. В результате выполнения ГБ НИР № 0120.0 704719 разработан экспериментальный вариант программной системы контроля знаний на основе тестовых заданий открытой формы. Разработанное программное обеспечение может быть использовано в отделах при приёме на работу новых сотрудников.

Основные функции, реализуемые программной системой:

- формирование базы языковых и предметных знаний, включая описание ситуации действительности семантически эквивалентными фразами естественного языка, выявление смыслового эталона ситуации употребления языка для описания факта действительности, формирование словаря синонимов и тезауруса предметной области;
- подготовка тестовых заданий и проведение тестирования, в том числе группового.

Разработанная программная система базируется на принципах, методах и алгоритмах, изложенных в диссертационной работе Михайлова Д.В.

2. Анализ функциональных возможностей программной системы, проведённый в рамках её тестирования при приёме на работу молодых специалистов в ОАО “НИИПТ “Растр” показал, что разработанное программное обеспечение может выполнять функции интеллектуального партнёра менеджера по персоналу на предприятии.

3. На основании выше изложенного считаем целесообразным внедрение подобных систем на межотраслевом уровне, их развитие и адаптацию к задачам, решаемым на предприятиях конкретной отрасли.

От НовГУ

_____ А.Л. Гавриков

_____ Г.М. Емельянов

От ОАО “НИИПТ “Растр”

Начальник отделения 2
_____ Я.Ю. Гозман

Ведущий научный сотрудник отделения 2
_____ Н.И. Смирнов

УТВЕРЖДАЮ
Проректор по НР НовГУ
_____ Бондаренко Е.А.
_____ 2012 г.

А К Т

об апробации результатов НИР
в учебном процессе

Мы, нижеподписавшиеся, зав. кафедрой ИТиС Гавриков А.Л. и научный руководитель темы 410-ИТиС/гр (гос. рег. № 0120.1 164263) Емельянов Г.М. составили настоящий акт о нижеследующем:

1. Согласно плану работ по теме 410-ИТиС/гр разработан опытный вариант программной системы контроля знаний на основе тестовых заданий открытой формы. Разработанное программное обеспечение может быть использовано для тестирования знаний студентов высших и средних учебных заведений.

В состав функций программной системы входит формирование базы языковых знаний и тезауруса предметной области на основе описания ситуаций действительности семантически эквивалентными фразами естественного языка, а также подготовка тестового материала и проведение тестирования. При разработке архитектуры программной системы были использованы теоретические и практические результаты, полученные в диссертации Михайлова Д.В.

2. Тестирование системы в ходе рубежной и итоговой аттестации студентов по дисциплине “Распознавание образов и обработка изображений” весеннего семестра 2011/2012 учебного года показало, что разработанное программное обеспечение позволяет автоматизировать подготовку тестового материала для организации федерального Internet-тестирования. Реализованная программная система представлена в открытом доступе на www.machinelearning.ru.

3. На основании выше изложенного считаем целесообразным расширенное использование подобных систем в масштабах Минобрнауки России, их развитие и адаптацию к задачам формирования общекультурных и профессиональных компетенций у бакалавров и магистров в рамках современных ФГОС ВПО.

Зав. кафедрой ИТиС НовГУ
_____ А.Л. Гавриков

Научный руководитель темы 410-ИТиС/гр
_____ Г.М. Емельянов