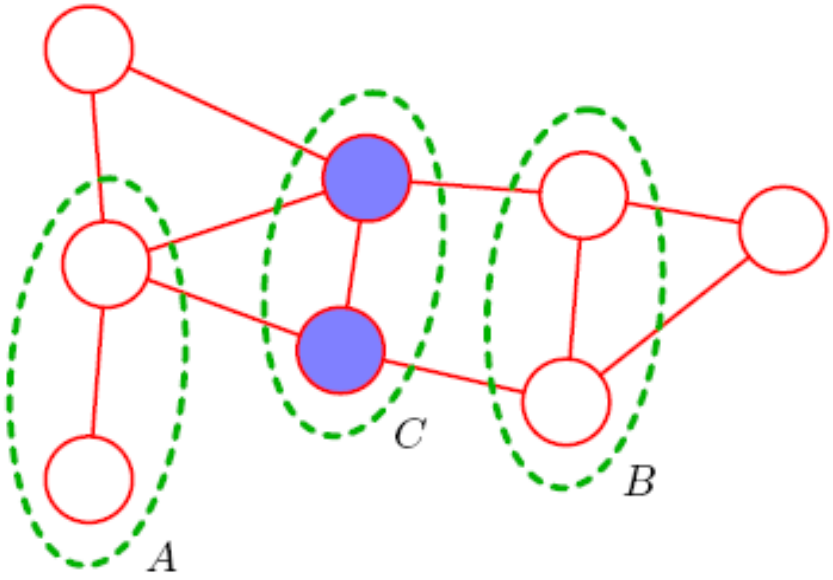
Лекция 4. Алгоритмы обмена сообщениями. Методы настройки потенциалов случайных полей.

Спецкурс «Структурные методы анализа изображений и сигналов»

Ольга Баринова

Моделирование зависимостей при помощи графических моделей

- Марковская сеть:
  - задается ненаправленным графом, ребра которого связывают переменные, между которыми есть непосредственные (а не опосредованные) зависимости



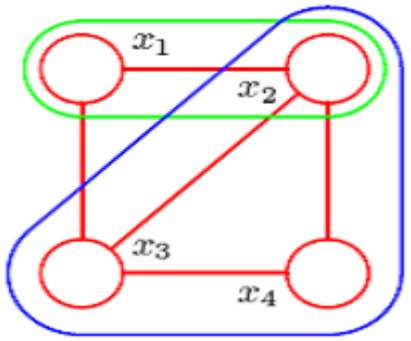
Моделирование зависимостей при помощи графических моделей

В общем виде совместное распределение значений элементов сети записывается с помощью неотрицательных потенциальных функций, определенных на максимальных кликах:

$$p(X) = \frac{1}{Z}\prod_{C}\psi_C(X_C) \qquad Z = \sum_{X}\prod_{C}\psi_C(X_C), \quad \psi_C(X_C) \geq 0$$

На рисунке синяя клика является максимальной, а зеленая – нет.
Совместное распределение имеет вид:

$$p(X) = \frac{1}{Z}\psi_1(x_1, x_2, x_3)\psi_2(x_2, x_3, x_4)$$



Моделирование зависимостей при помощи графических моделей

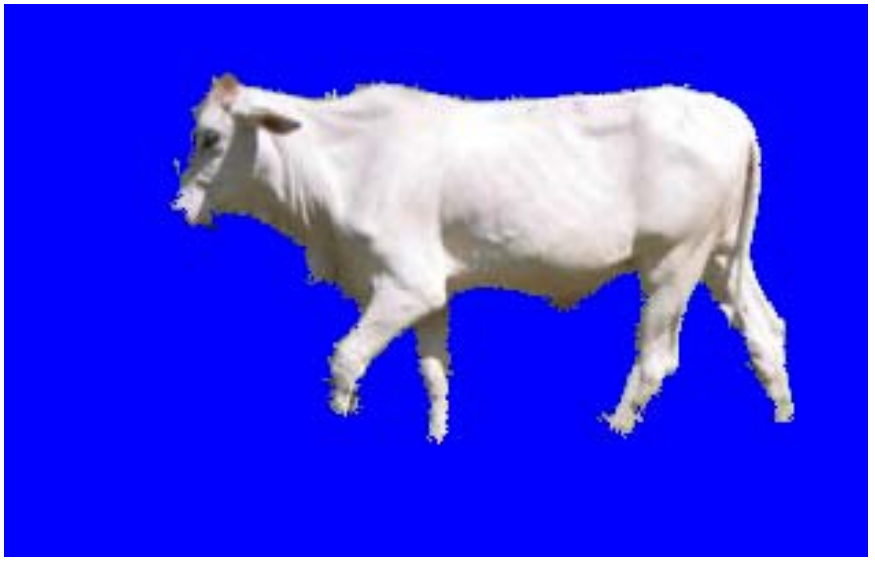
•Для многих приложений требуется находить наиболее вероятное состояние системы:

$$\arg\max p(X) = \arg\max \frac{1}{Z}\prod_C \psi_C(X_C) =$$

$$= \arg\max p(X) = \arg\max \exp\left(-\sum_C E_C(X_C)\right) =$$

$$= \arg\min \sum_C E_C(X_C)$$

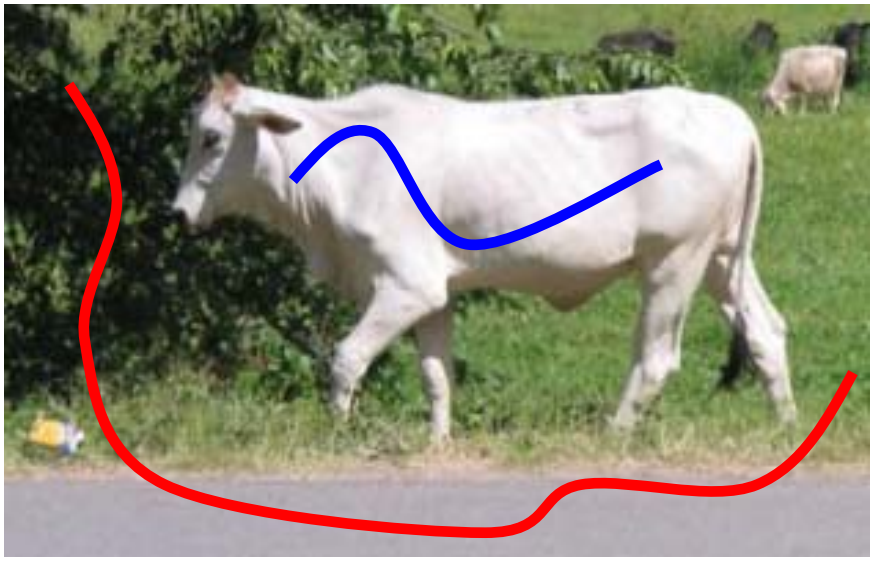
Интерактивная сегментация

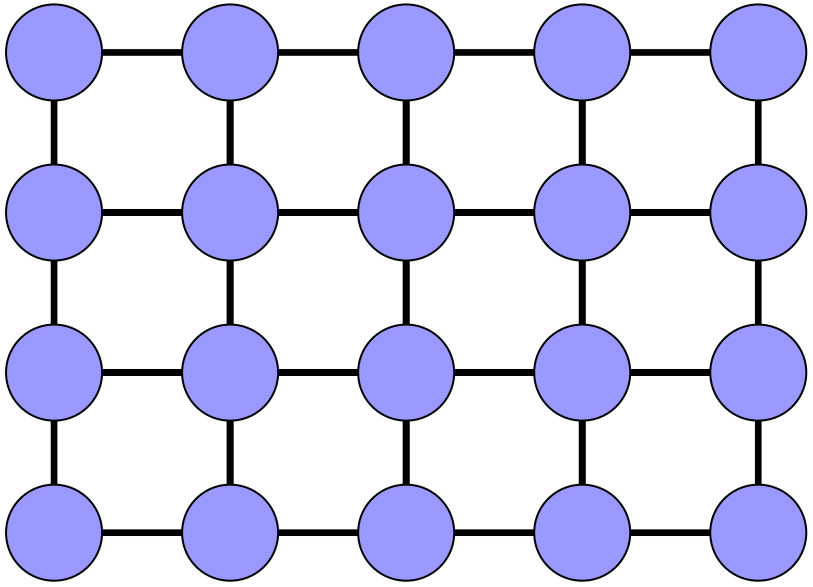


Как ?

Моделируем знания об изображении при помощи марковской сети

Интерактивная сегментация





Граф скрытых переменных G = (V,E)

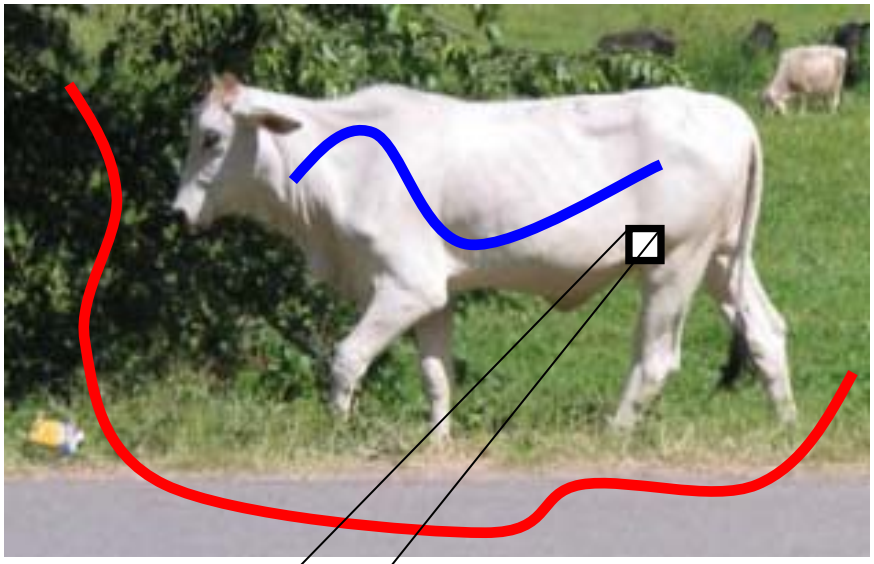
Каждая вершина соответствует скрытой переменной

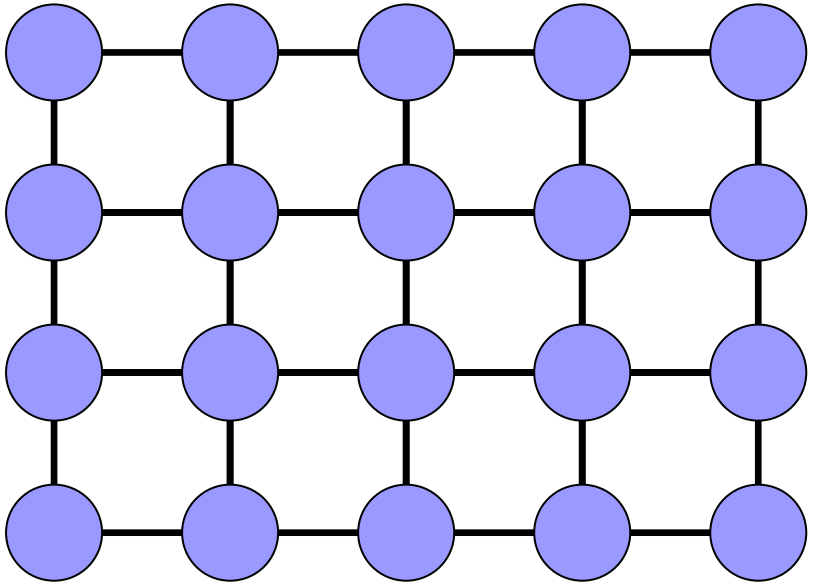
Их столько же, сколько пикселей

Ребра соединяют вершины в соответствии с 4-связностью

Скрытые переменные принимают значения L = {obj,bkg}

Интерактивная сегментация

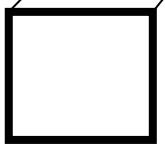




Граф скрытых переменных G = (V,E)

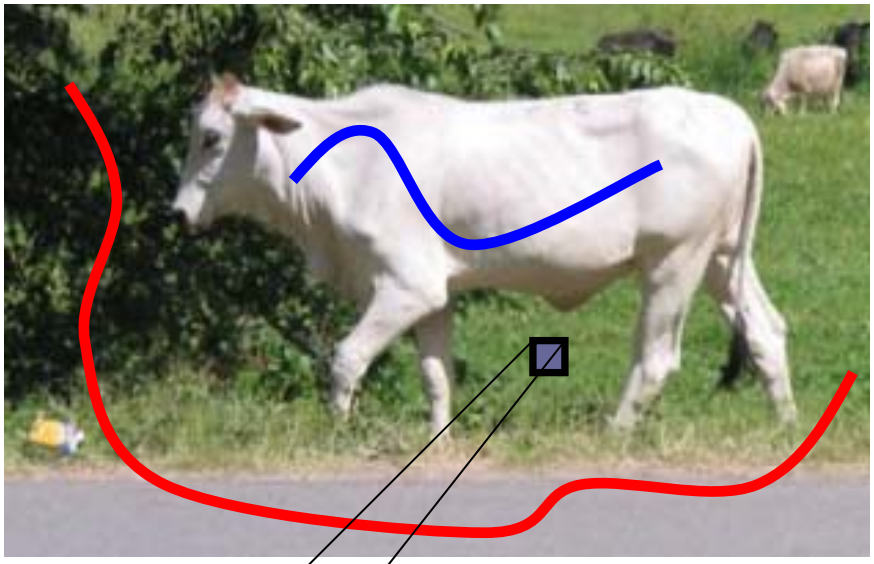
Объект белый, фон – серый/зеленый

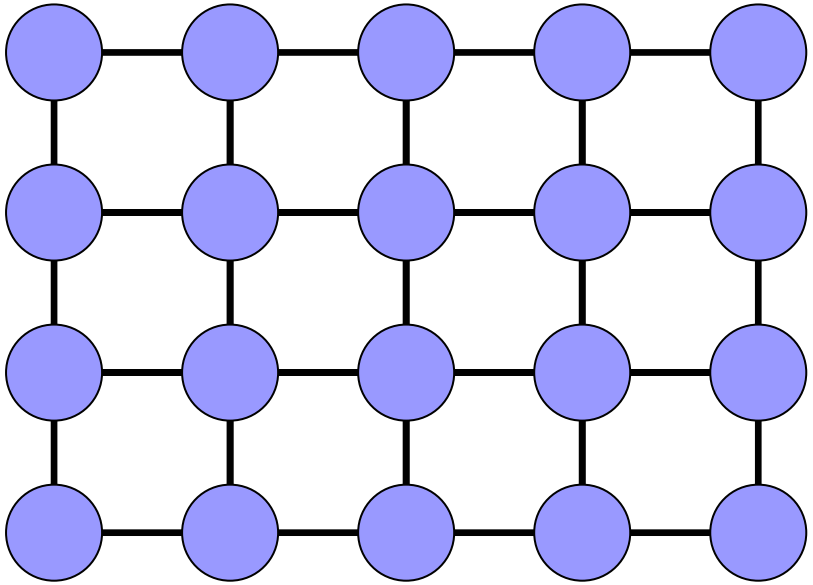
Унарный потенциал



Вероятность значения ‘obj’ большая
Вероятность значения ‘bkg’ маленькая

Интерактивная сегментация





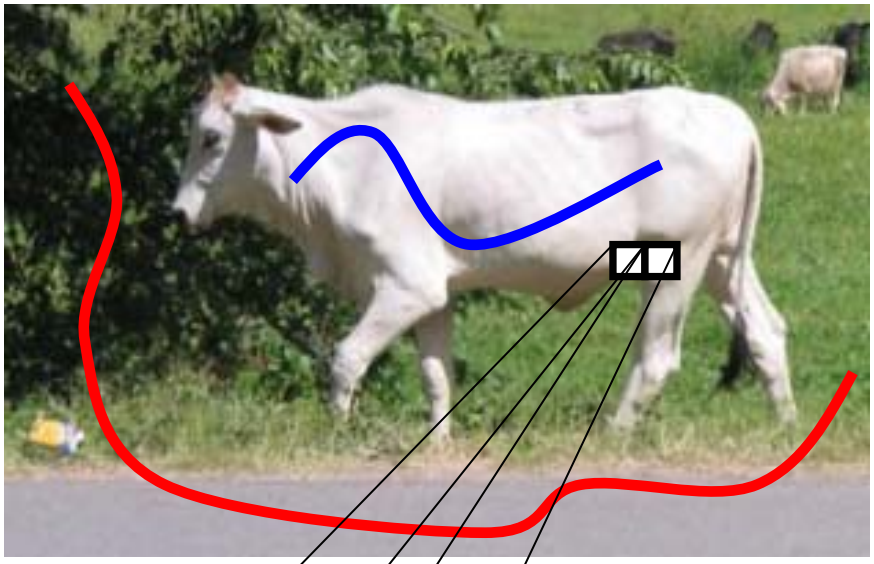
Граф скрытых переменных G = (V,E)

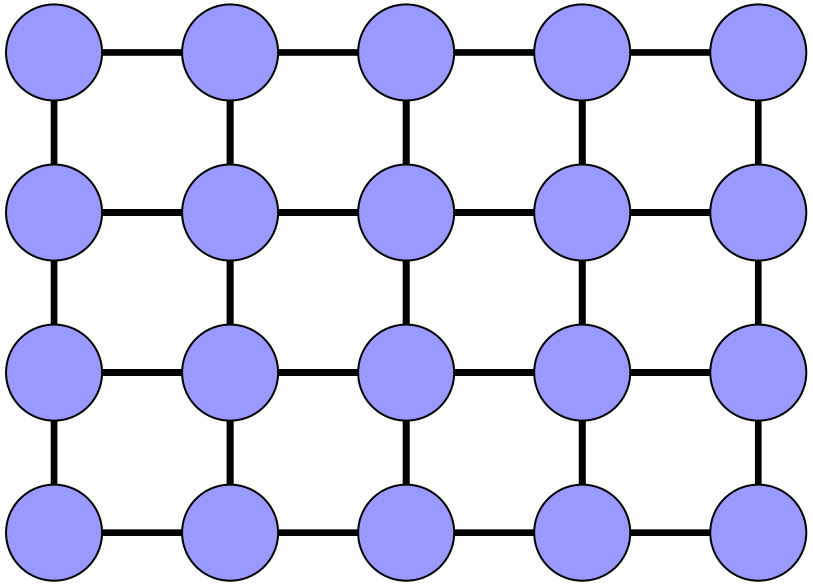
Объект белый, фон – серый/зеленый

Унарный потенциал

Вероятность значения ‘obj’ маленькая
Вероятность значения ‘bkg’ большая

Интерактивная сегментация

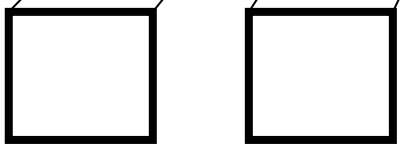




Граф скрытых переменных G = (V,E)

Объект белый, фон – серый/зеленый

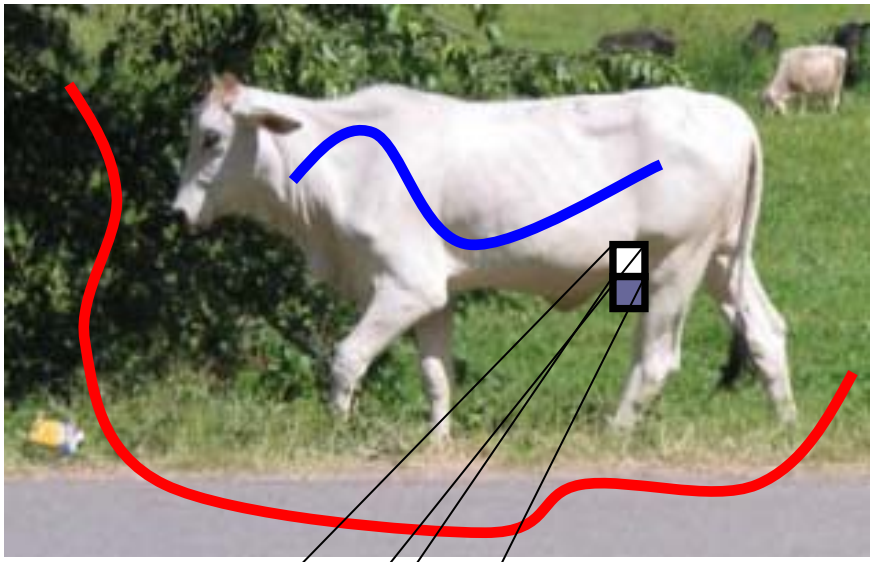
Парный потенциал

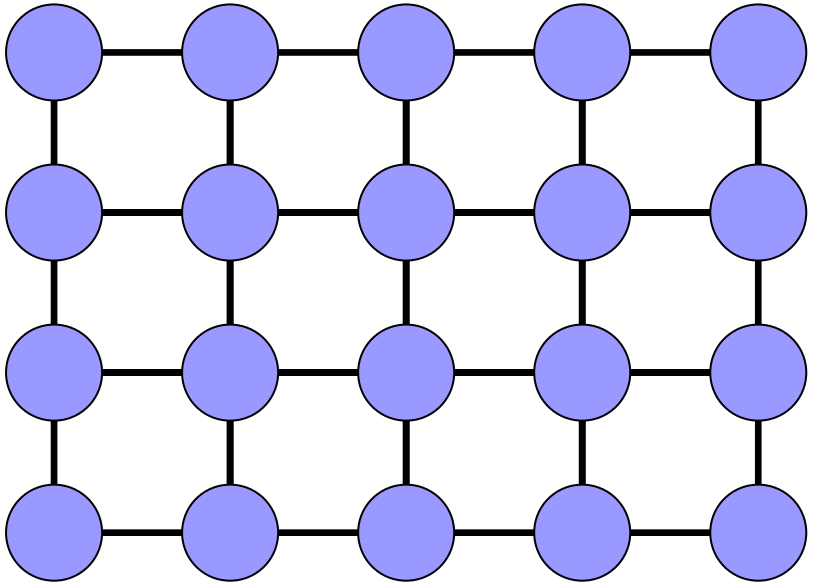


Вероятность одинаковых меток большая

Вероятность разных меток маленькая

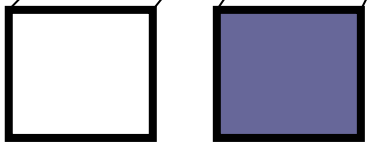
Интерактивная сегментация





Граф скрытых переменных G = (V,E)

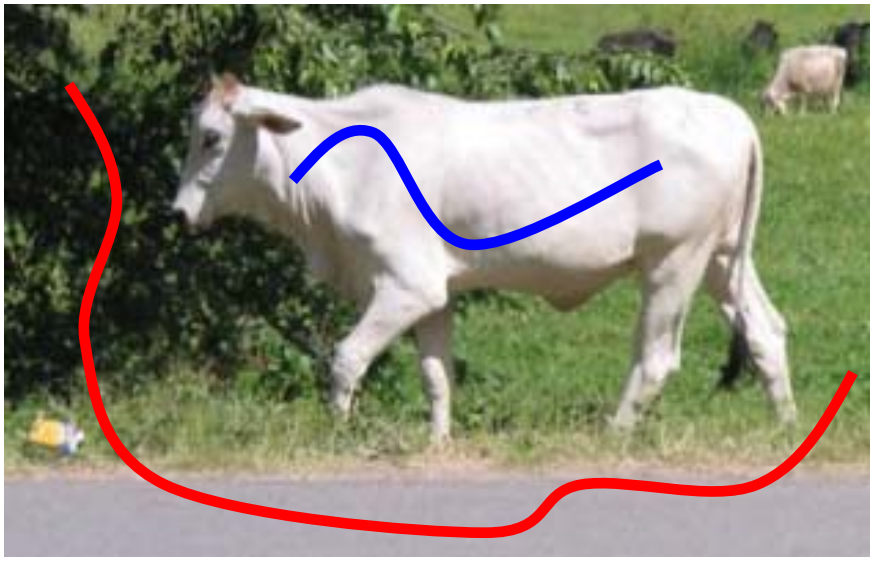
Парный потенциал



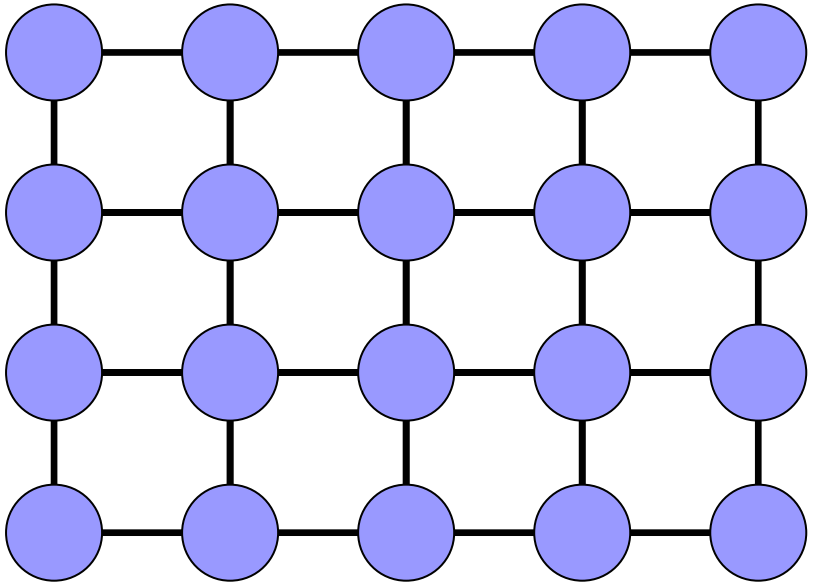
Вероятность одинаковых меток маленькая

Вероятность разных меток большая

Интерактивная сегментация



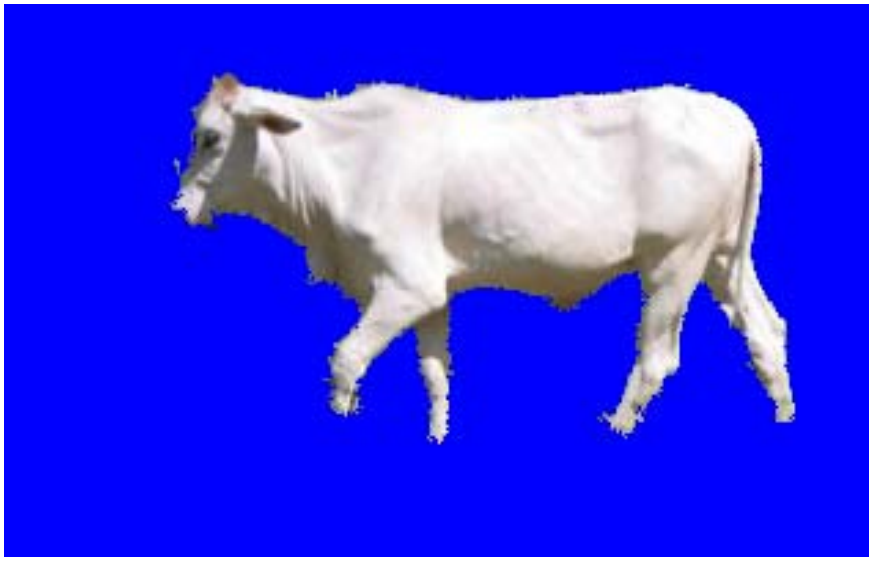
Объект белый, фон – серый/зеленый



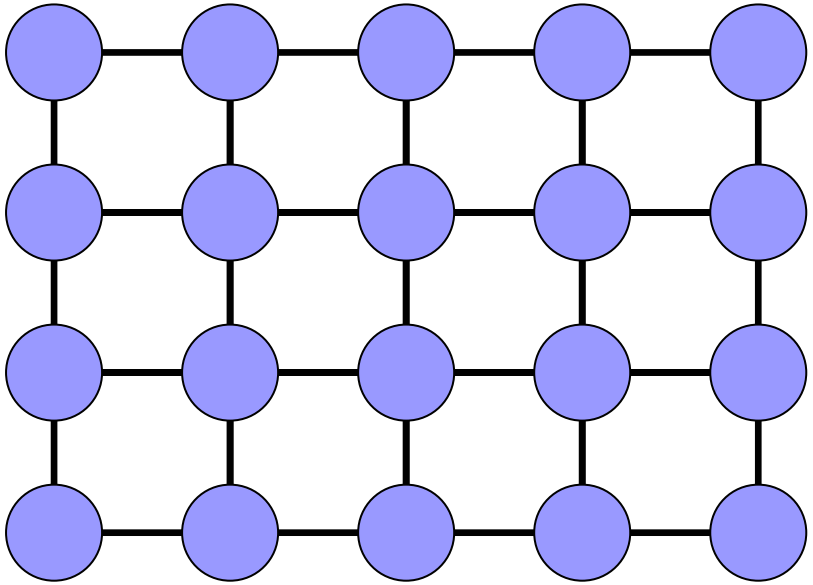
Граф скрытых переменных G = (V,E)

Сегментация = минимизация энергии

Интерактивная сегментация



Объект белый, фон – серый/зеленый



Граф скрытых переменных G = (V,E)

Сегментация = минимизация энергии

Методы вывода в графических моделях

- Задача нахождения наиболее вероятной разметки графа (минимизации энергии) в общем случае NP-трудная
- Все методы нахождения наиболее вероятной разметки графа в общем случае дают приближенное решение

- Для частных случаев графических моделей и некоторых специальных видов функции энергии существуют точные методы нахождения минимума энергии
- Вычислительно эффективные методы для нахождения минимума функции энергии также существуют не для всех графических моделей и не для всех видов энергии

Какую энергию можно минимизировать с помощью разрезов графов?

- **Опр.** Пусть $E(x_1,....,x_n)$ - функция n двоичных переменных, и пусть I, J - непересекающееся разбиение множества индексов $\{1,....n\}$: $I = \{i(1),...,i(m)\}$, $J = \{j(1),...., j(n-m)\}$. Пусть $\alpha_{i(1)},....,\alpha_{i(m)}$ - какие-то двоичные константы. Проекцией $E' = E[x_{i(1)} = \alpha_{i(1)},...,x_{i(m)} = \alpha_{i(m)}]$ называется функция $n-m$ переменных, определенная как

$$E'(x_{j(1)},....,x_{j(n-m)}) = E(x_1,...,x_n)$$

где $x_i = \alpha_i$ для $i \in I$. Мы будем говорить, что зафиксировали переменные $x_{i(1)},.....,x_{i(m)}$.

Какую энергию можно минимизировать с помощью разрезов графов?

- **Опр.** (регулярной функции):
  - Все функции одной переменной являются регулярными.
  - Функция E двух переменных называется регулярной, если $E(0,0) + E(1,1) \leq E(0,1) + E(1,0)$.
  - Функция E более чем двух переменных называется регулярной, если все проекции E двух переменных являются регулярными.

- **Опр.** Назовем функцию энергии E граф-представимой, если для нее можно построить граф, минимальный разрез в котором будет соответствовать минимуму данной энергии.

Какую энергию можно минимизировать с помощью разрезов графов?

- **Теорема 1.** Пусть E - функция n двоичных переменных, которая может быть представлена в виде суммы

$$E(x_1,...,x_n) = \sum_i E^i(x_i) + \sum_{i<j} E^{i,j}(x_i,x_j) + \sum_{i<j<k} E^{i,j,k}(x_i,x_j,x_k)$$

  Тогда E является граф-представимой тогда и только тогда, когда E является регулярной.

- **Теорема 2**. Пусть E функция двоичных переменных. Если E не является регулярной, то E не граф-представима.

Итерационные алгоритмы для минимизации энергии (1)

- Общий вид энергии:

$$E(x_1,...,x_n) = \sum_i E^i(x_i) + \sum_{i,j} E^{i,j}(x_i, x_j)$$

$$x_i \in \{1,...K\} \qquad K > 2$$

$$\arg\min_{x_1,...,x_n} E(x_1,...,x_n) - ?$$

Итерационные алгоритмы (2)

- **Опр. 1** Пусть α ,β - метки. Изменение разбиения P (маркировки X) на новое разбиение P' (маркировку X) называется $\alpha - \beta$ заменой, если $P_l = P'_l$ для любой метки $l \neq \alpha, \beta$.

- **Опр. 2** Пусть имеется метка α . Изменения разбиения P (маркировки X) на новое разбиение P' (маркировку F') называется α -расширением, если $P_\alpha \subset P'_\alpha$ и $P'_l \subset P_l$ для любой метки $l \neq \alpha$.

Итерационные алгоритмы: α-β-замена

- **Алгоритм, основанный на $\alpha - \beta$ замене.**

  1. Начать с произвольной маркировки X
  2. Успех := 0
  3. Для каждой пары меток $\{\alpha, \beta\} \subset L$
     a. Найти $\hat{X} = \arg\min_{X} E(X)$ среди всех X', являющихся $\alpha - \beta$ заменой X
     b. Если $E(\hat{X}) < E(X)$, установить $X := \hat{X}$ и Успех := 1
  4. Если Успех = 1, goto 2
  5. Вернуть X

- $E^{i,j}$ должна быть полу-метрикой.

Итерационные алгоритмы: α-расширение

- **Алгоритм, основанный на α –расширении.**

1. Начать с произвольной маркировки X
2. Успех := 0
3. Для каждой пары меток $\{\alpha, \beta\} \subset L$
   a. Найти $\hat{X} = \arg\min_X E(X)$ среди всех X', являющихся α – расширением
   b. Если $E(\hat{X}) < E(X)$, установить $X := \hat{X}$ и Успех := 1
4. Если Успех = 1, goto 2
5. Вернуть X

- $E^{i,j}$ должна быть полу-метрикой.

Методы вывода в графических моделях

- Для решения практической задачи обычно требуется сравнить результаты разных методов минимизации энергии и выбрать наиболее подходящий по скорости/качеству/классу функций энергии

- При выборе структуры графической модели и функции энергии надо иметь в виду, каким методом планируется минимизировать энергию

Вопросы?

План лекции

- Алгоритмы обмена сообщениями
- Условное случайное поле
- Методы настройки потенциалов случайных полей

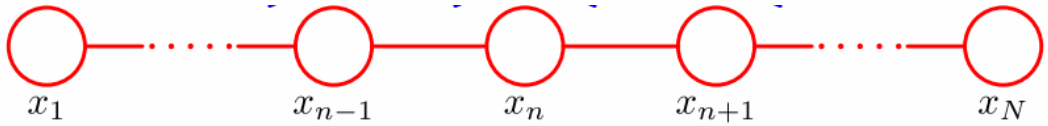
План лекции

- Алгоритмы обмена сообщениями

- Условное случайное поле

- Методы настройки потенциалов случайных полей

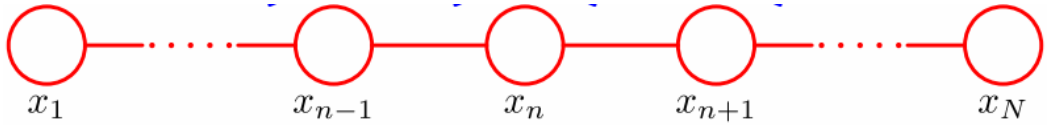
Граф-цепочка



- Рассмотрим граф-цепочку
- Функция совместного распределения имеет вид

$$p(\mathbf{x}) = \frac{1}{Z}\psi_{1,2}(x_1, x_2)\psi_{2,3}(x_2, x_3)\cdots\psi_{N-1,N}(x_{N-1}, x_N)$$

Вычисление частных распределений для цепочки

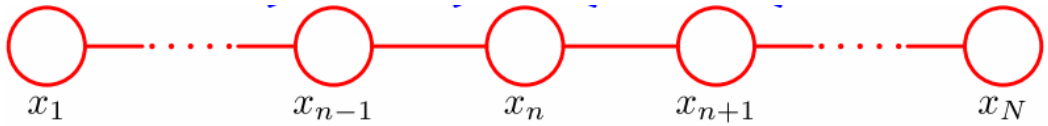


- Рассмотрим задачу вычисления распределений отдельных переменных
- Распределение отдельной переменной x_n вычисляется по формуле

$$p(x_n) = \sum_{x_1} \cdots \sum_{x_{n-1}} \sum_{x_{n+1}} \cdots \sum_{x_N} p(\mathbf{x})$$

- Сложность вычислений растет экспоненциально с увеличением длины цепочки

Вычисление частных распределений для цепочки



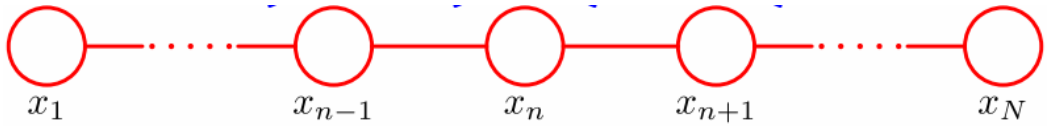
• Объем вычислений можно сократить, за счет использования информации об условной независимости переменных

•Рассмотрим суммирование по x_N. От нее зависит только потенциал $\psi_{N-1,N}(x_{N-1}, x_N)$. Произведя суммирование по x_N получим функцию от x_{N-1}

$$\sum_{x_N} \psi_{N-1,N}(x_{N-1}, x_N)$$

•Далее продолжим производить суммирование по последующим переменным вдоль цепочки – принцип динамического программирования

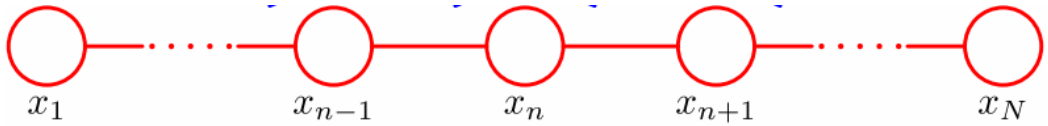
Вычисление частных распределений для цепочки



• Если таким образом сгруппировать слагаемые в сумме, получим выражение, сложность вычисления которого $O(NK^2)$

$$p(x_n) = \frac{1}{Z} \underbrace{\left[\sum_{x_{n-1}} \psi_{n-1,n}(x_{n-1}, x_n) \cdots \left[\sum_{x_2} \psi_{2,3}(x_2, x_3) \left[\sum_{x_1} \psi_{1,2}(x_1, x_2)\right]\right] \cdots\right]}_{\mu_\alpha(x_n)} \underbrace{\left[\sum_{x_{n+1}} \psi_{n,n+1}(x_n, x_{n+1}) \cdots \left[\sum_{x_N} \psi_{N-1,N}(x_{N-1}, x_N)\right] \cdots\right]}_{\mu_\beta(x_n)}$$

Вычисление частных распределений для цепочки



• Таким образом, распределение вероятностей для x_n представимо в виде произведения сообщений от соседей

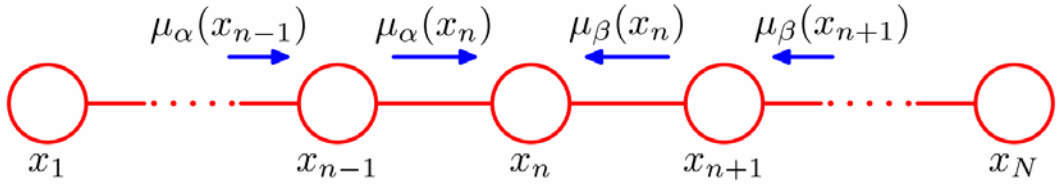
$$p(x_n) = \frac{1}{Z}\mu_\alpha(x_n)\mu_\beta(x_n)$$

•Сообщения вычисляются рекуррентно

$$\mu_\alpha(x_2) = \sum_{x_1} \psi_{1,2}(x_1, x_2) \qquad \mu_\alpha(x_n) = \sum_{x_{n-1}} \psi_{n-1,n}(x_{n-1}, x_n) \left[\sum_{x_{n-2}} \cdots \right]$$

$$= \sum_{x_{n-1}} \psi_{n-1,n}(x_{n-1}, x_n)\mu_\alpha(x_{n-1})$$

Алгоритм sum-product. Случай графа-цепочки.



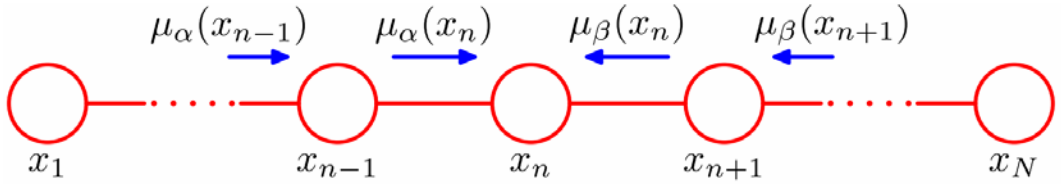
• Теперь предположим, что требуется вычислить распределение для всех переменных, входящих в цепочку.

•Наивный алгоритм: вычислять распределение для каждой переменной отдельно. Многие сообщения будут вычисляться по несколько раз.

•Алгоритм sum-product:

- •передать сообщения из правого конца цепочки в левый конец
- •передать сообщение из левого конца цепочки в правый конец
- •в результате вычислены частные распределения для каждой переменной

Алгоритм sum-product. Случай графа-цепочки.

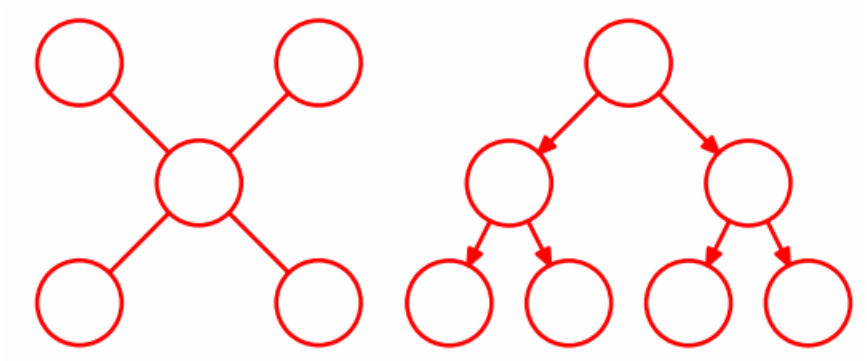


• Для max-product требуется всего в 2 раза больше вычислений по сравнению с вычислением распределения для одной переменной, а не в N раз больше, как в случае наивного алгоритма

•Сложность алгоритма линейная по числу переменных в цепочке

•Сообщение – вектор длины *K*, иначе говоря в каждое значение переменной приходит свое сообщение

Графы древовидной структуры



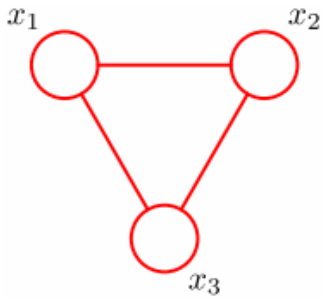
• Дерево – граф, в котором между любой парой вершин существует не более одного пути

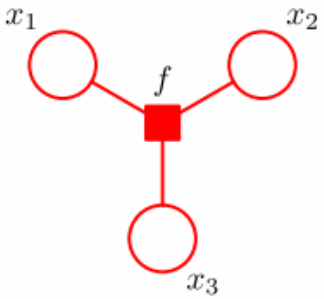
•Деревья не содержат циклов

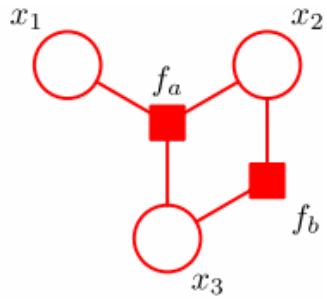
•В случае направленных графов морализация не приводит к появлению дополнительных ребер

Фактор-граф

$$f(x_1, x_2, x_3) = \psi(x_1, x_2, x_3)$$







$$f_a(x_1, x_2, x_3) f_b(x_1, x_2) = \psi(x_1, x_2, x_3)$$

• Вероятность представляется произведением некоторых факторов

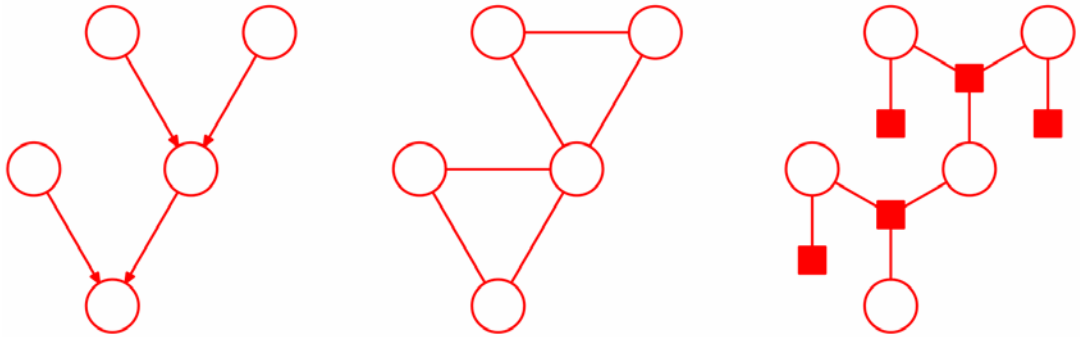
$$p(\mathbf{x}) = \prod_s f_s(\mathbf{x}_s)$$

•Фактор-граф

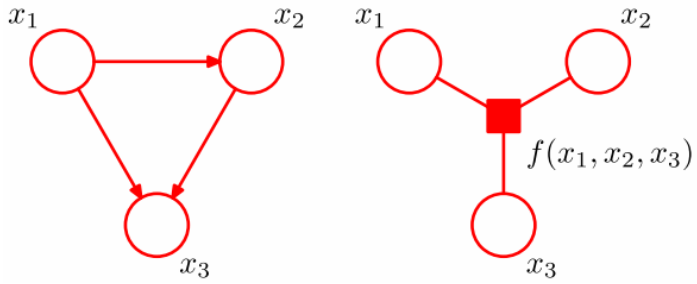
•двудольный граф, состоящих из вершин, соответствующим случайным величинам и вершин, соответствующих факторам

•факторы присоединены к соответствующим вершинам

Фактор-граф



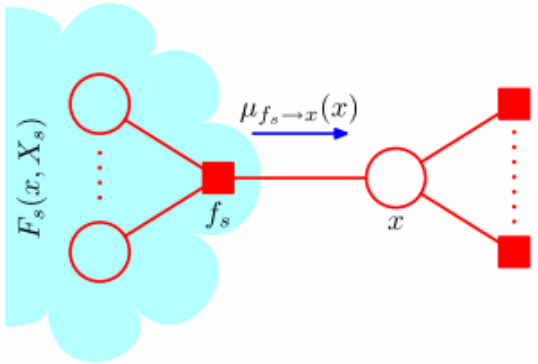
• Если исходный граф содержит циклы, то можно построить соответствующий ему фактор-граф, который циклов не содержит



Вычисление частных распределений для деревьев

• Частное распределение вычисляется по формуле $p(x) = \sum_{\mathbf{x} \setminus x} p(\mathbf{x})$

•Группируем факторы, соседние с вершиной x $p(\mathbf{x}) = \prod_{s \in \mathrm{ne}(x)} F_s(x, X_s)$

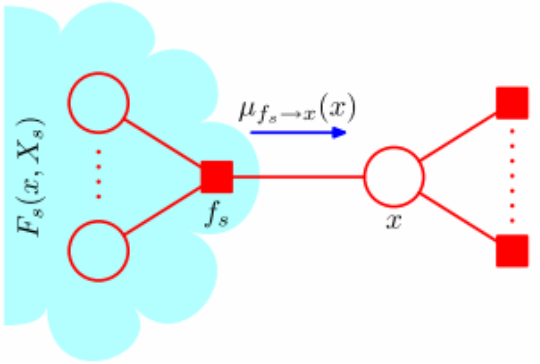


- $\mathrm{ne}(x)$ - множество всех факторов, соседних с вершиной *x*
- X_s - множество всех переменных, соединенных с *x* через фактор f_s
- $F_s(x, X_s)$ - произведение всех факторов, входящих в группу

Вычисление частных распределений для деревьев

• Поменяем местами суммы и произведения

$$p(x) = \prod_{s \in \mathrm{ne}(x)} \left[\sum_{X_s} F_s(x, X_s) \right]$$
$$= \prod_{s \in \mathrm{ne}(x)} \mu_{f_s \to x}(x).$$



Сообщение от фактора к переменной

•Раскроем выражение для сообщения

$$F_s(x, X_s) = f_s(x, x_1, \ldots, x_M) G_1(x_1, X_{s1}) \ldots G_M(x_M, X_{sM})$$

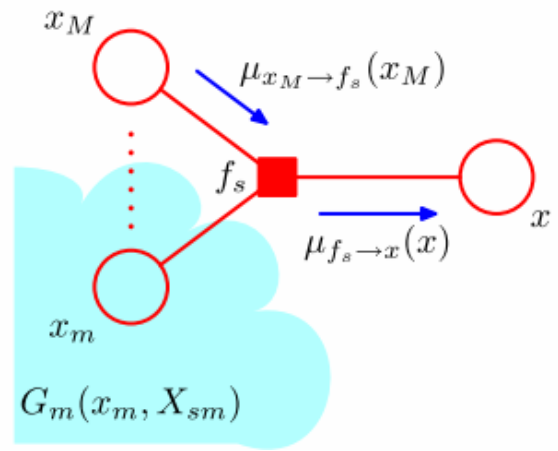
• $\{x, x_1, \ldots, x_M\}$ - группа переменных, от которых зависит фактор

Вычисление частных распределений для деревьев

•Каждая группа факторов представляет собой поддерево, поэтому можно вывести рекуррентные формулы для вычисления сообщений

$$\mu_{f_s \to x}(x) = \sum_{x_1} \cdots \sum_{x_M} f_s(x, x_1, \ldots, x_M) \prod_{m \in \mathrm{ne}(f_s) \setminus x} \left[\sum_{X_{xm}} G_m(x_m, X_{sm}) \right]$$

$$= \sum_{x_1} \cdots \sum_{x_M} f_s(x, x_1, \ldots, x_M) \prod_{m \in \mathrm{ne}(f_s) \setminus x} \mu_{x_m \to f_s}(x_m)$$



Сообщение от переменной к фактору

Вычисление частных распределений для деревьев

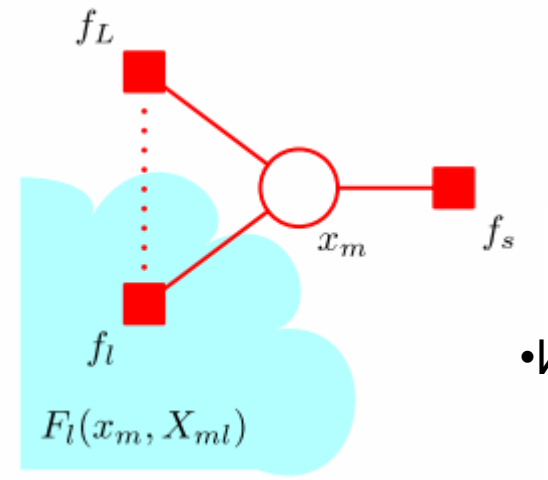
• Сообщение от переменной к фактору выражается следующим образом

$$G_m(x_m, X_{sm}) = \prod_{l \in \mathrm{ne}(x_m) \setminus f_s} F_l(x_m, X_{ml})$$

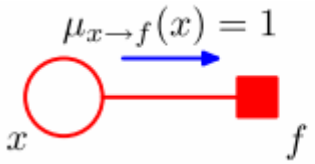
•Подставим выражение в формулу для $F_l(x_m, X_{ml})$

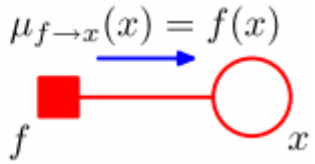
$$\mu_{x_m \to f_s}(x_m) = \prod_{l \in \mathrm{ne}(x_m) \setminus f_s} \left[\sum_{X_{ml}} F_l(x_m, X_{ml}) \right]$$

$$= \prod_{l \in \mathrm{ne}(x_m) \setminus f_s} \mu_{f_l \to x_m}(x_m)$$



•Инициализация:





Алгоритм sum-product. Случай графов древовидной структуры

- Вычисление частного распределения для переменной *x*
  - Примем вершину *x* за корень дерева. Передадим сообщения из листов дерева к корню и обратно
  - Пользуясь рекуррентными формулами для вычисления сообщений от факторов к переменным и от переменных к факторам, вычислим сообщения, пришедшие в вершину x
  - Частное распределение для переменной x получается по формуле

$$p(x) = \prod_{s \in \mathrm{ne}(x)} \mu_{f_s \to x}(x)$$

Алгоритм sum-product. Случай графов древовидной структуры

- Теперь предположим, что требуется найти частные распределения для всех переменных, входящих в граф

- Алгоритм sum-product
  - Примем произвольную вершину за корень
  - Передадим сообщения от листов к корню
  - Передать сообщения от корня к листам
  - Вычислить частные распределения как произведения сообщений от факторов

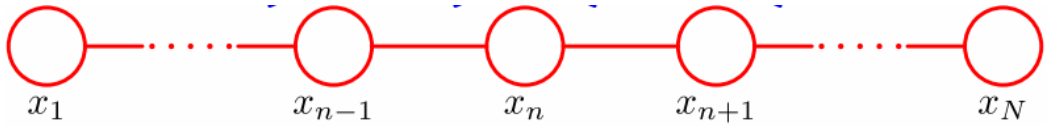
Алгоритм sum-product. Случай графов древовидной структуры

•На выходе получается точное решение для частных распределений каждой переменной

•Требуется всего в 2 раза больше вычислений, чем для подсчета одного частного распределения

•Алгоритм имеет линейную сложность от числа вершин в графе

Вычисление наиболее вероятной разметки для цепочки



• Теперь вернемся к задаче нахождения наиболее вероятной разметки. Требуется найти

$$\mathbf{x}^{\max} = \arg\max_{\mathbf{x}} p(\mathbf{x})$$

•Иначе говоря

$$p(\mathbf{x}^{\max}) = \max_{\mathbf{x}} p(\mathbf{x})$$

•Требуется найти эффективный алгоритм вычисления

$$\max_{\mathbf{x}} p(\mathbf{x}) = \max_{x_1} \ldots \max_{x_M} p(\mathbf{x})$$

Вычисление наиболее вероятной разметки для цепочки

•Сокращение вычислений основано на следующих правилах:

$$\max(a+b, a+c) = a + \max(b, c)$$

$$\max(ab, ac) = a \max(b, c)$$

$$\ln\left(\max_{\mathbf{x}} p(\mathbf{x})\right) = \max_{\mathbf{x}} \ln p(\mathbf{x})$$

•Выражение максимума совместной вероятности записывается как

$$\max_{\mathbf{x}} p(\mathbf{x}) = \frac{1}{Z} \max_{x_1} \cdots \max_{x_N} \left[\psi_{1,2}(x_1, x_2) \cdots \psi_{N-1,N}(x_{N-1}, x_N)\right]$$

$$= \frac{1}{Z} \max_{x_1} \left[\psi_{1,2}(x_1, x_2) \left[\cdots \max_{x_N} \psi_{N-1,N}(x_{N-1}, x_N)\right]\right].$$

Алгоритм max-sum. Случай графов древовидной структуры.

•По аналогии с алгоритмом sum-product получим формулы для сообщений

$$\mu_{f\to x}(x) = \max_{x_1,\ldots,x_M}\left[\ln f(x,x_1,\ldots,x_M) + \sum_{m\in \mathrm{ne}(f_s)\setminus x} \mu_{x_m\to f}(x_m)\right]$$

$$\mu_{x\to f}(x) = \sum_{l\in \mathrm{ne}(x)\setminus f} \mu_{f_l\to x}(x).$$

•Сообщения инициализируются

$$\mu_{x\to f}(x) = 0$$

$$\mu_{f\to x}(x) = \ln f(x)$$

Алгоритм max-sum. Случай графов древовидной структуры.

- Алгоритм max-sum
  - Примем произвольную вершину за корень
  - Передадим сообщения от листов к корню, а затем от корня к листам
  - Наиболее вероятная разметка графа определяется по формуле

$$x^{\max} = \arg\max_{x} \left[\sum_{s \in \mathrm{ne}(x)} \mu_{f_s \to x}(x) \right]$$

Алгоритмы max-sum и sum-product. Случай графов с циклами.

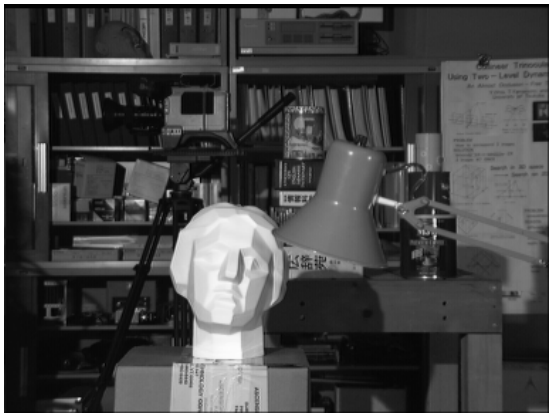
- Сообщения от переменных к факторам и от факторов к переменным вычисляются локально
  - используется информация только о соседних вершинах
- Т. о. можно вычислять сообщения и для графов с циклами, в таком случае сообщения к одной и той же вершине могут передаваться несколько раз
- Положим, что каждый раз, когда новое сообщение приходит к вершине, оно заменяет собой старое сообщение, пришедшее в эту вершину
  - Также можно смешивать старое сообщение с новым (брать линейную комбинацию)

Алгоритмы max-sum и sum-product. Случай графов с циклами.

- Требуется задать расписание доставки сообщений
  - Flooding schedule – сообщения передаются по каждому ребру графа в обоих направлениях
  - Serial schedule – в каждый момент времени передается только одно сообщение
- Во многих задачах алгоритм дает хорошие решения
  - Однако в некоторых случаях зацикливается

Алгоритмы max-sum. Итоги

- Метод используется для получения наиболее вероятной разметки графа
- Метод дает точное решение для графов, имеющих структуру дерева
- Применим для произвольного вида энергии
- В случае графов с циклами дает приближенное решение



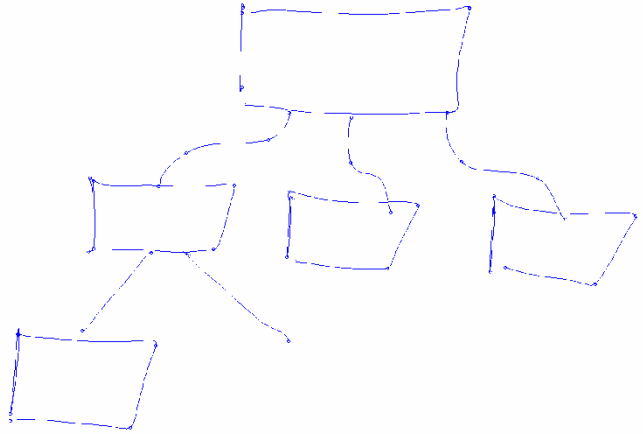
Вопросы?

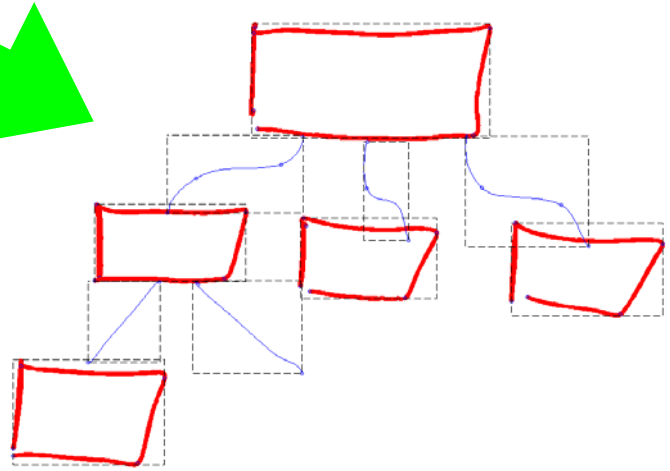
План лекции

- Алгоритмы обмена сообщениями
- Условное случайное поле
- Методы настройки потенциалов случайных полей

Задача: оцифровка схем

Схема нарисована от руки

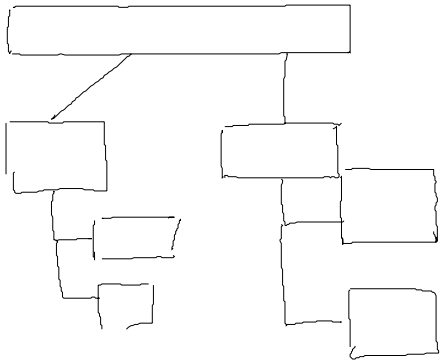




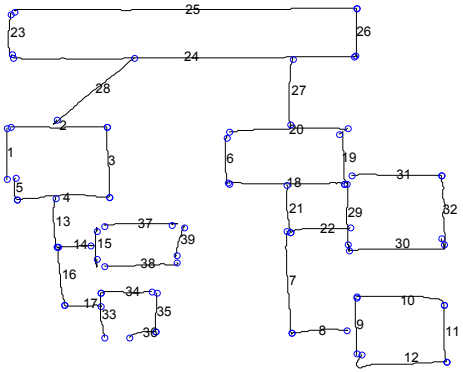
требуется ее оцифровать

Схема решения задачи

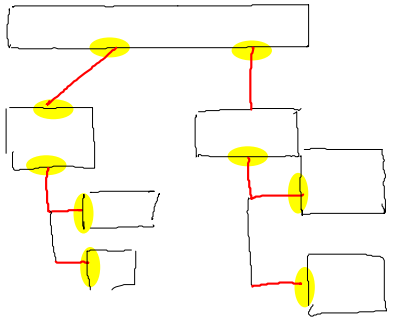
Входные данные



Выделить отдельные штрихи

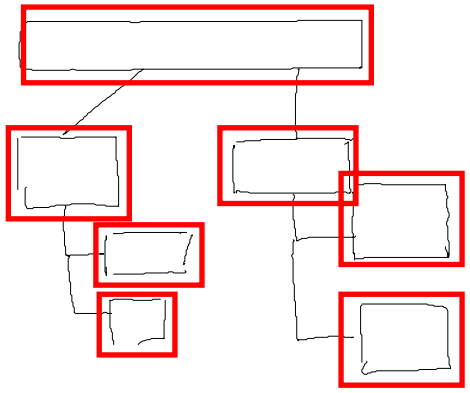


Для каждого штриха вычислить признаки



?

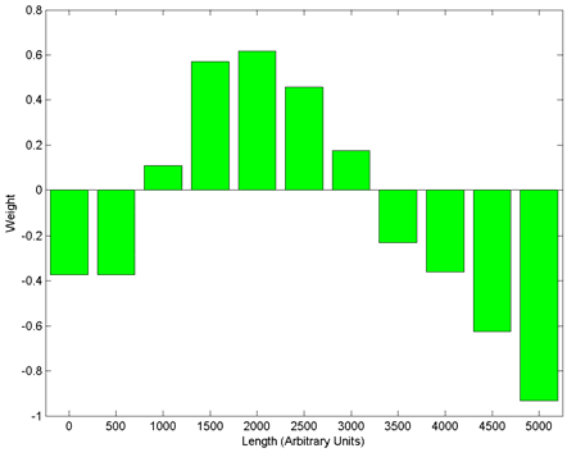
Разметить штрихи как контейнер или коннектор

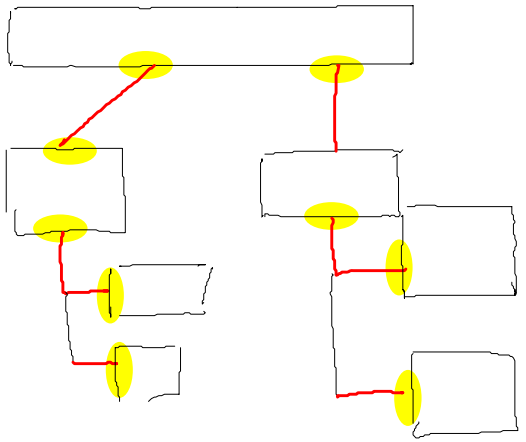


Заменить блоки стандартными фигурами

Признаки

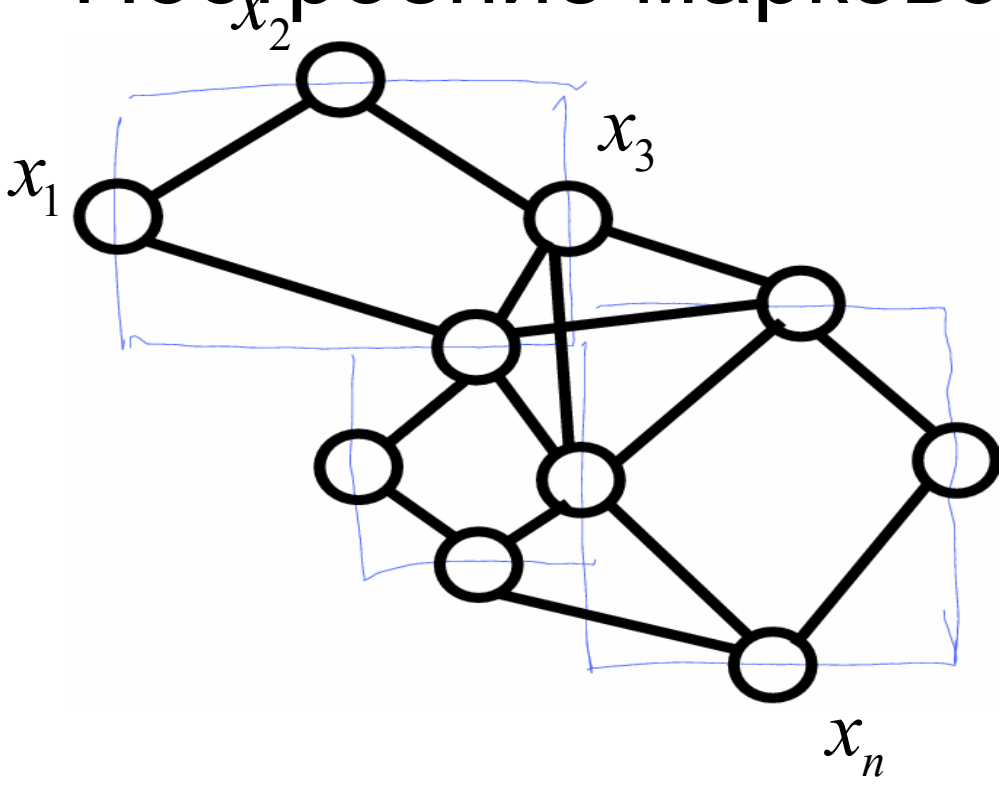
- Пространственные признаки
  - Гистограммы углов, длин, расстояний…
- Временные признаки
  - Фрагментов в одном штрихе, число несвязанных штрихов…
- Признаки связанные со взаимным расположением
  - Т-образное расположение…
- …





Т-образное расположение

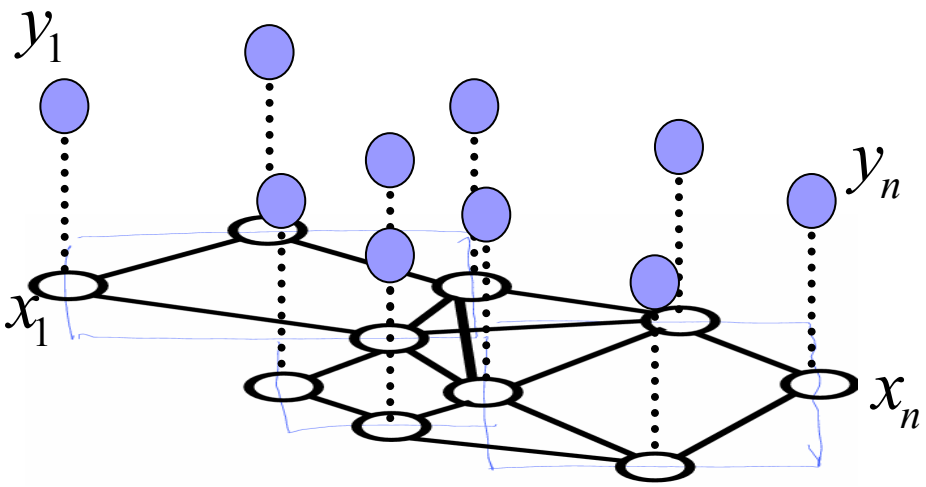
Построение марковской сети



•Считаем зависимыми только те штрихи, расстояние между которыми меньше определенного порога

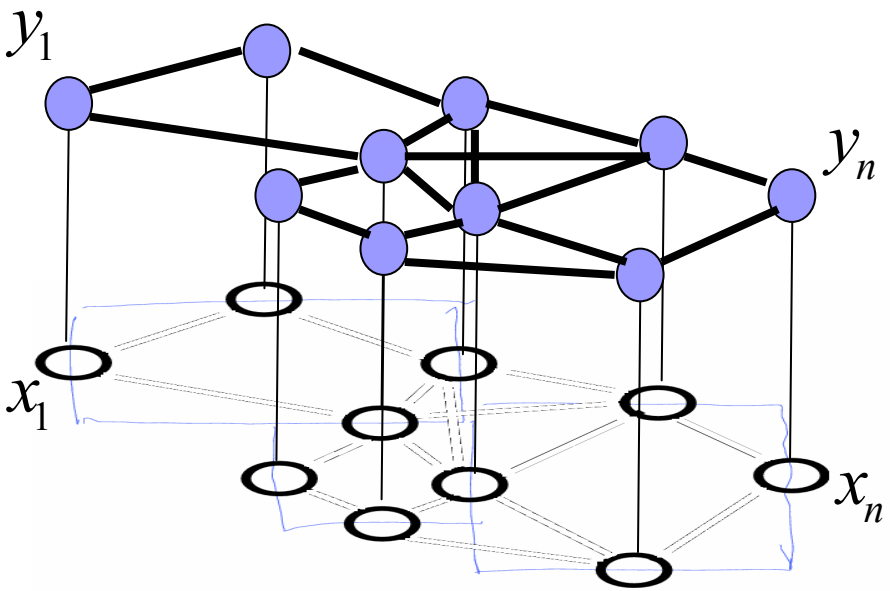
•Иначе говоря, считаем условно независимыми те штрихи, расстояние между которыми больше некоторого порога

Построение марковской сети



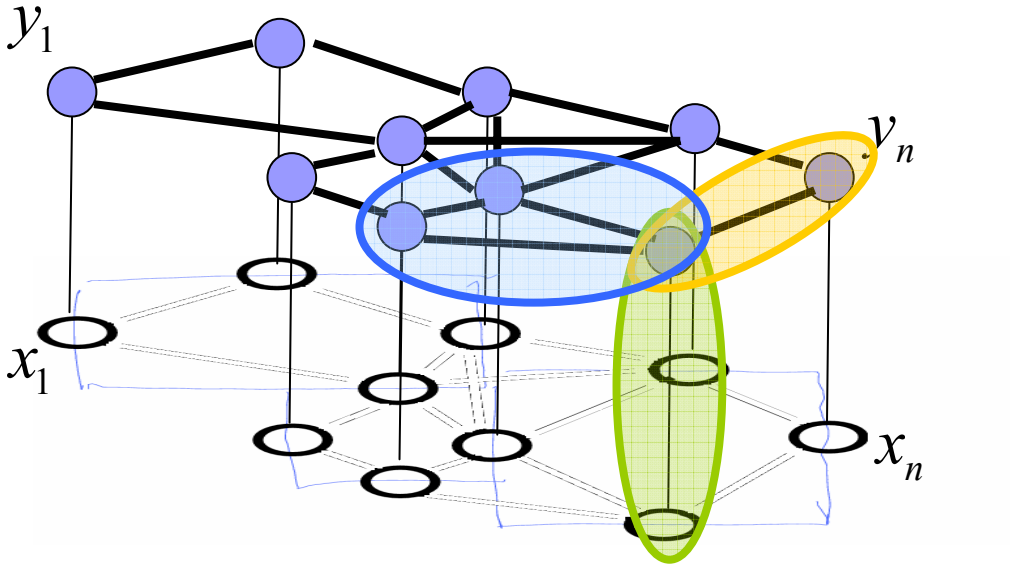
- Введем дополнительные вершины, соответствующие меткам штрихов – скрытые переменные
- Штрихам соответствуют наблюдаемые переменные

Построение марковской сети



- **Стандартный подход:**
  - Считаем зависимыми скрытые переменные, а каждая наблюдаемая переменная зависит только от соответствующей скрытой переменной

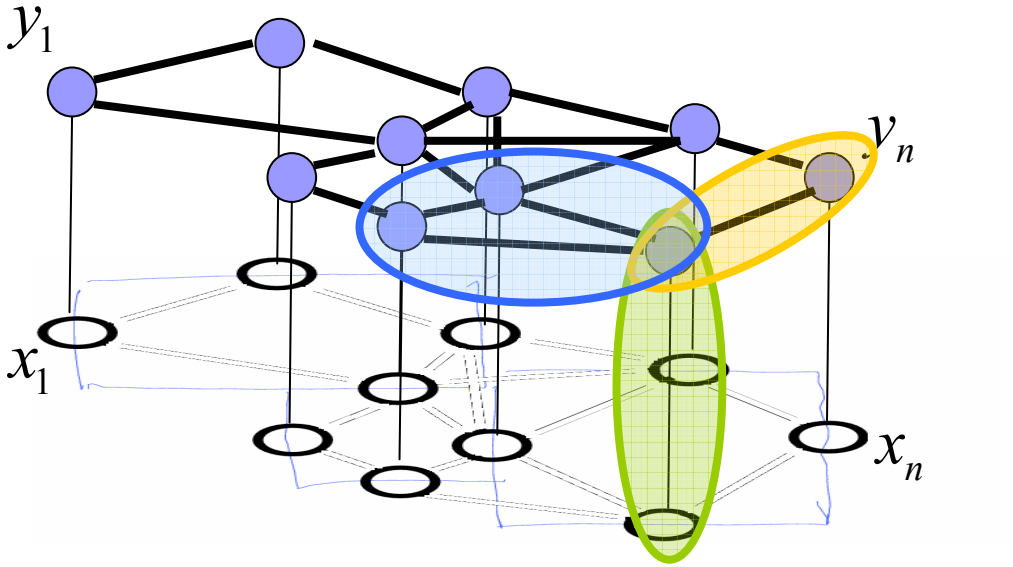
Построение марковской сети



- Структура такой марковской сети задает функцию распределения:

$$p(X,Y) = \frac{1}{Z}\prod_C \psi_C(X_C, Y_C)$$

Построение марковской сети

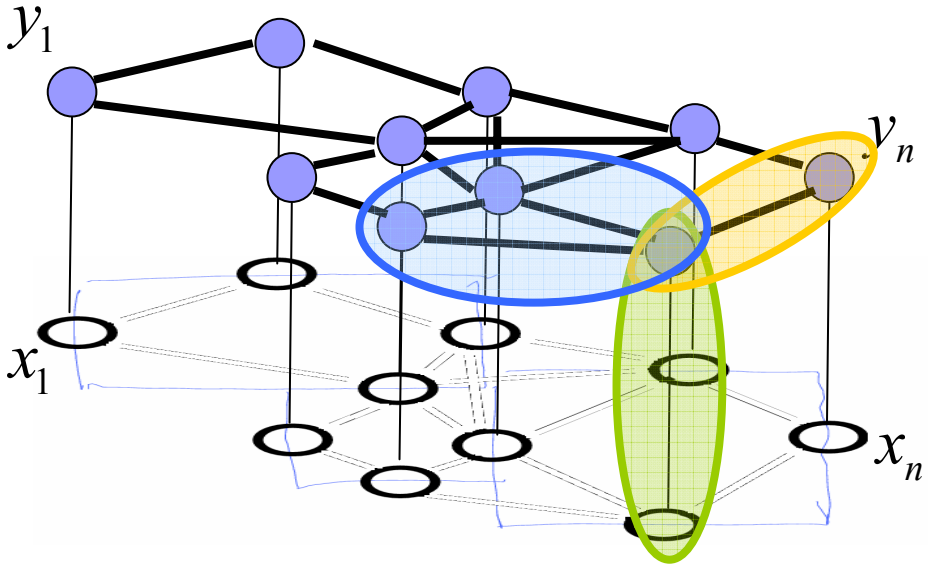


- Соответствующая функция энергии:

$$E(X,Y) = -\sum_C \ln \psi_C(X_C, Y_C)$$

- Наблюдаемые переменные входят только в двойные клики, то есть потенциалы высокого порядка не зависят от данных

Построение марковской сети

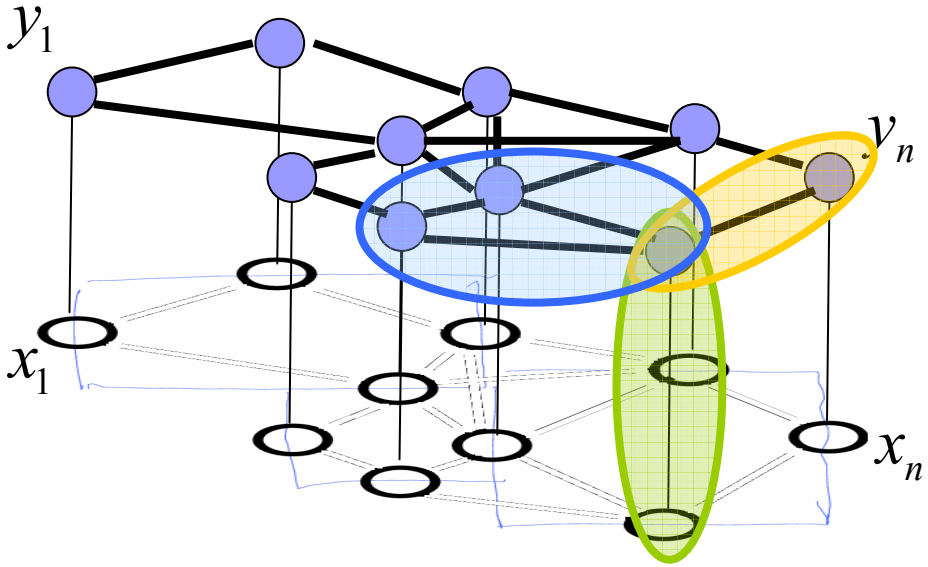


- **Недостатки такой модели:**
  - Предполагается условная независимость всех наблюдаемых переменных
  - Потенциалы клик высокого порядка зависят только от значений меток

- Поиск наиболее вероятных значений скрытых переменных:

$$E(X,Y) = -\sum_C \ln \psi_C(X_C, Y_C) \to \min_Y$$

Построение марковской сети



- Решение о том, какими должны быть значения скрытых переменных, принимается по принципу:

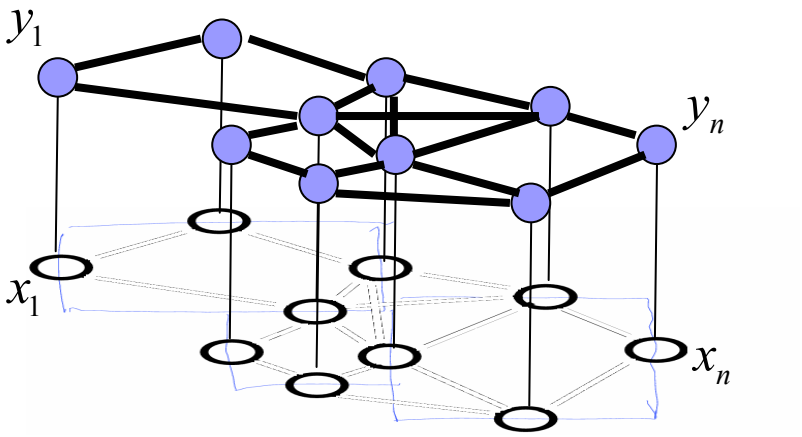
$$p(Y \mid X) \to \max_Y$$

Поскольку X фиксировано, достаточно найти Y:

$$p(Y, X) = p(Y \mid X)\, p(X) \to \max_Y$$

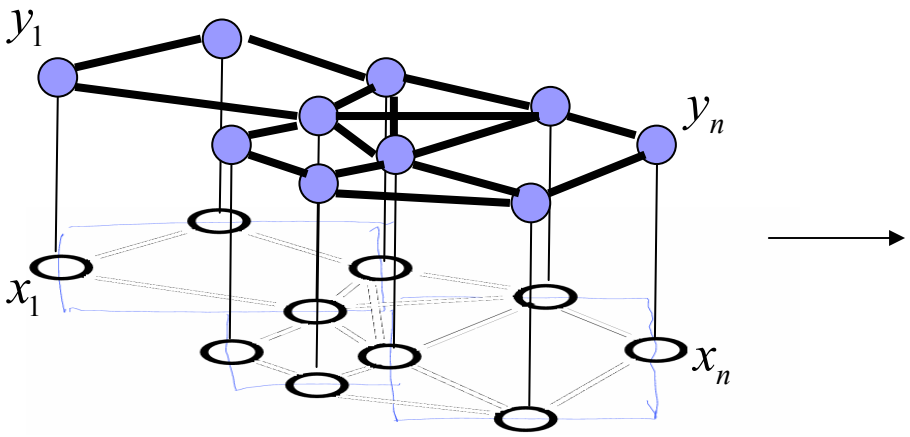
Условное случайное поле

- Условное распределение p(Y|X) моделируется напрямую



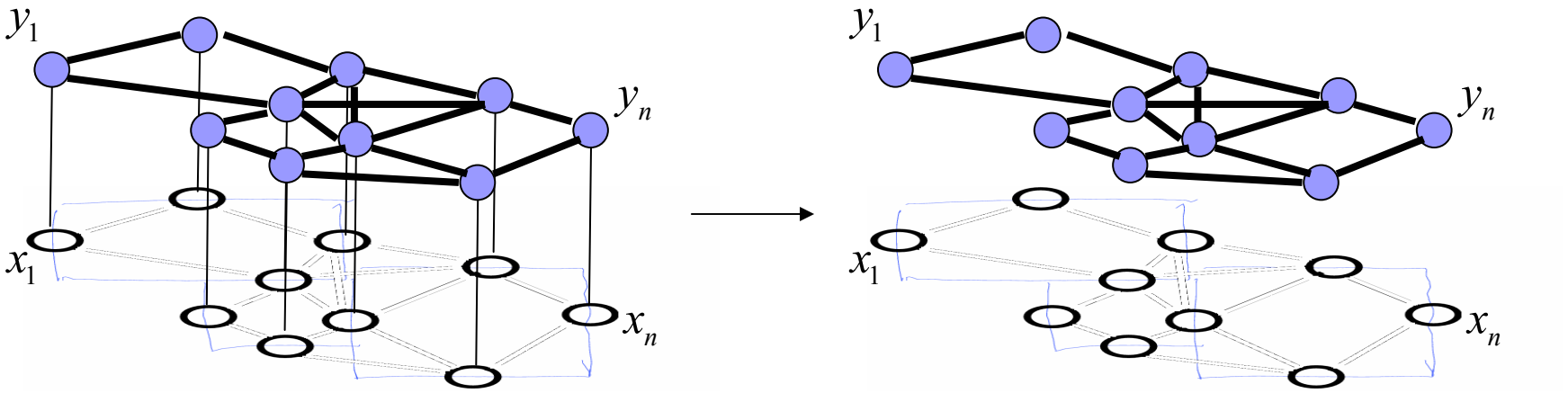
Условное случайное поле

- Условное распределение p(Y|X) моделируется напрямую



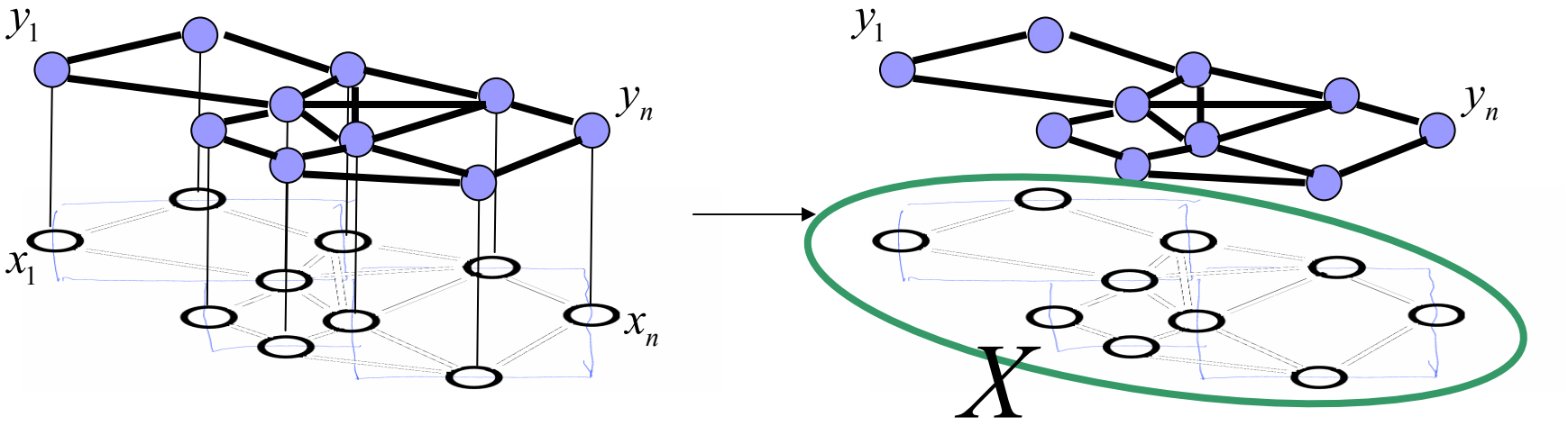
Условное случайное поле

- Условное распределение p(Y|X) моделируется напрямую



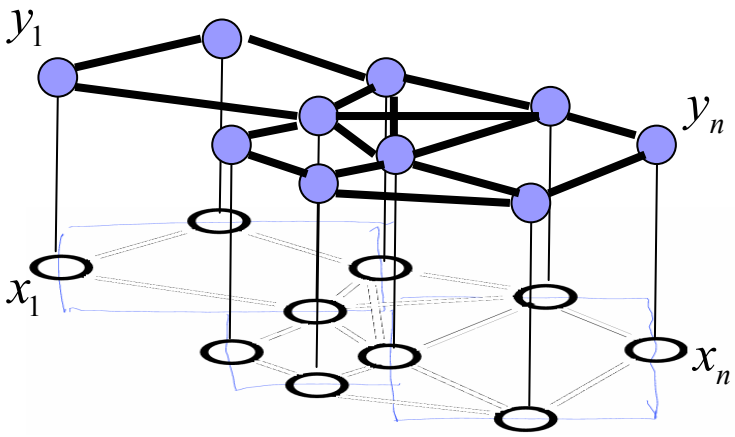
Условное случайное поле

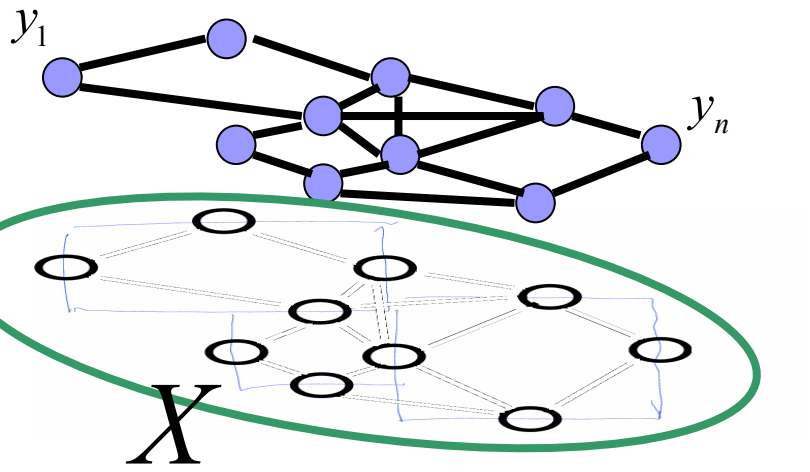
- Условное распределение p(Y|X) моделируется напрямую



Условное случайное поле

- Условное распределение p(Y|X) моделируется напрямую



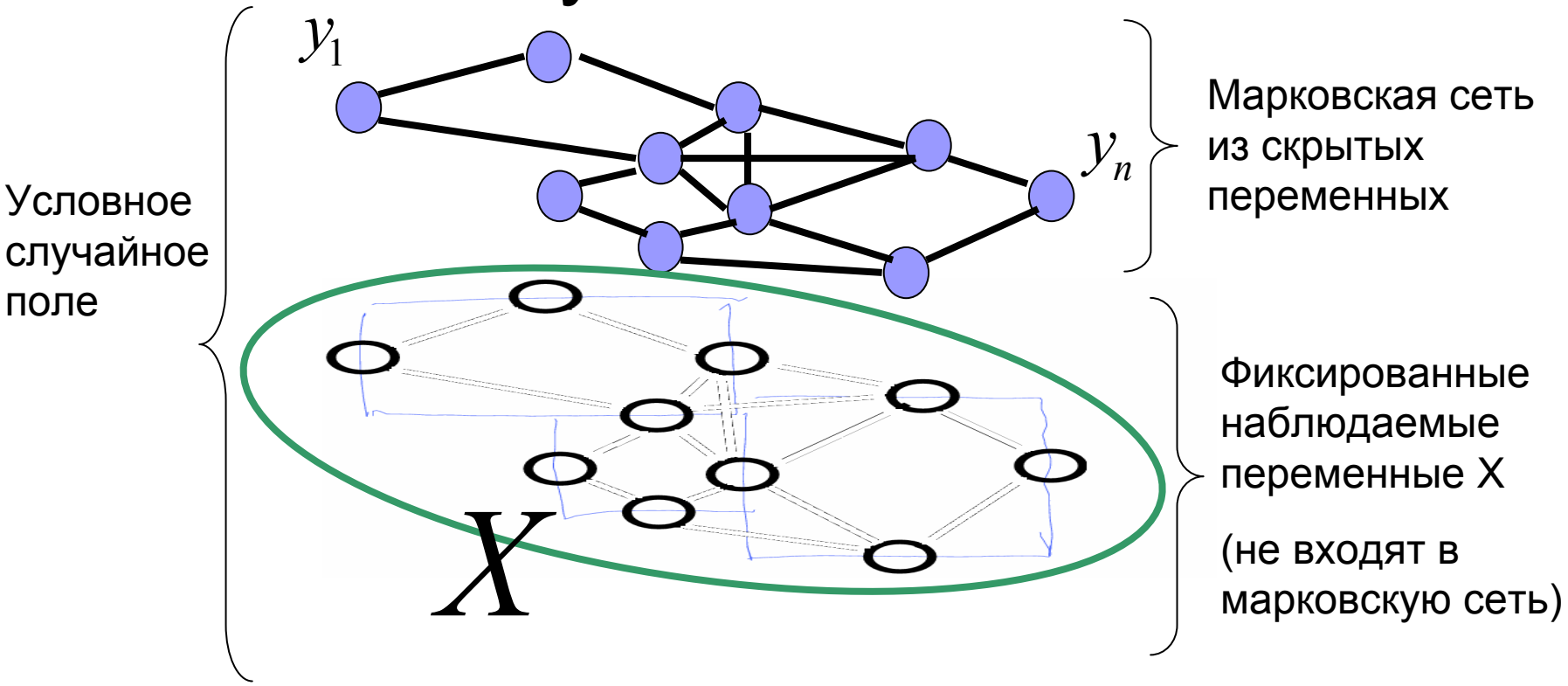


$$p(X,Y) = \frac{1}{Z}\prod_C \psi_C(X_C, Y_C)$$

$$p(Y \mid X) = \frac{1}{Z(X)}\prod_C \psi_C(Y_C \mid X)$$

$$Z(X) = \sum_{y \in \Upsilon}\prod_C \psi_C(y \mid X)$$

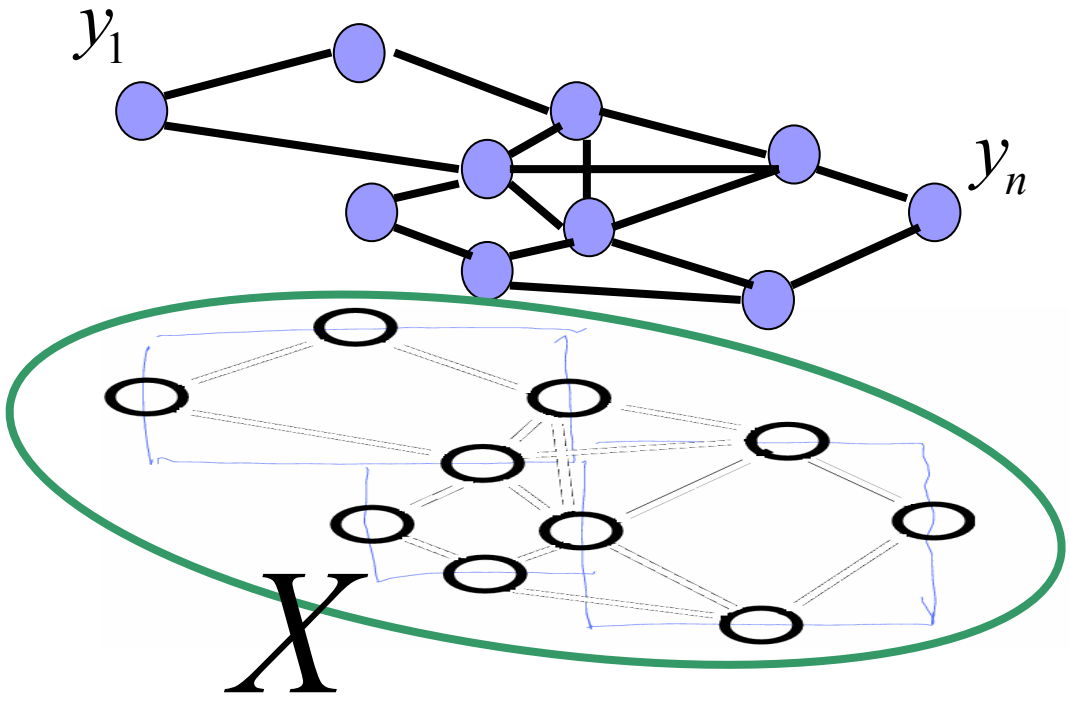
Условное случайное поле



- **Условное случайное поле:**

  марковская сеть, глобально обусловленная наблюдениями

Условное случайное поле



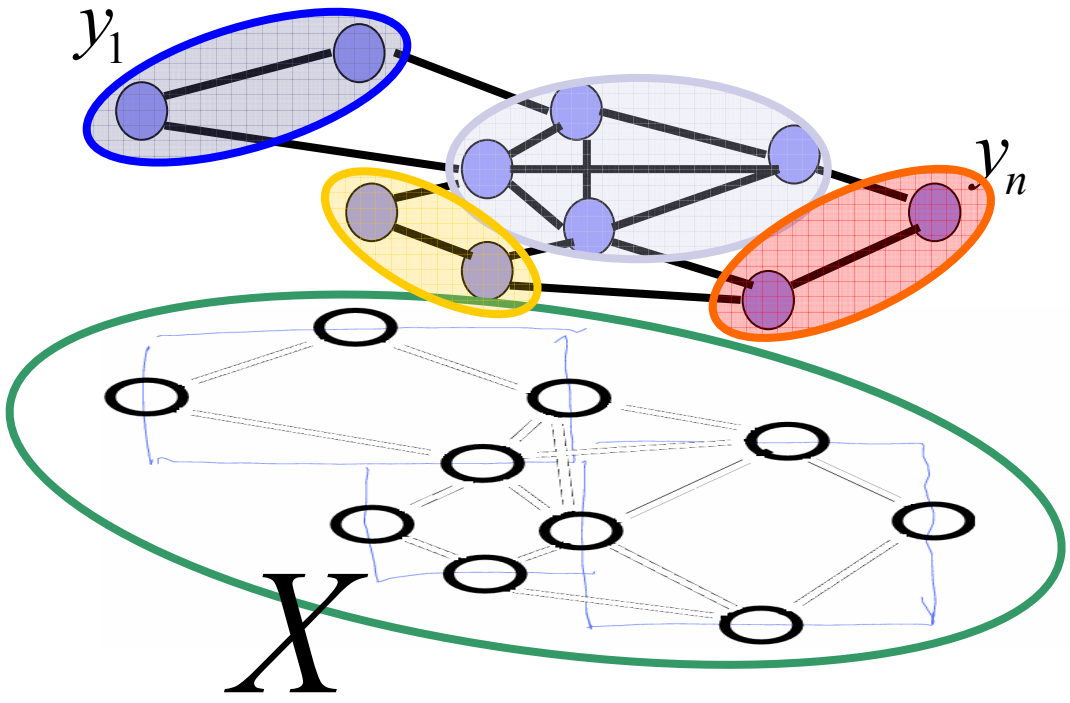
•Не моделируем лишних вещей, вроде распределения наблюдаемых величин

•Не предполагаем никаких зависимостей/независимостей между наблюдаемыми величинами

- **Условное случайное поле:**

  марковская сеть, глобально обусловленная наблюдениями

Условное случайное поле



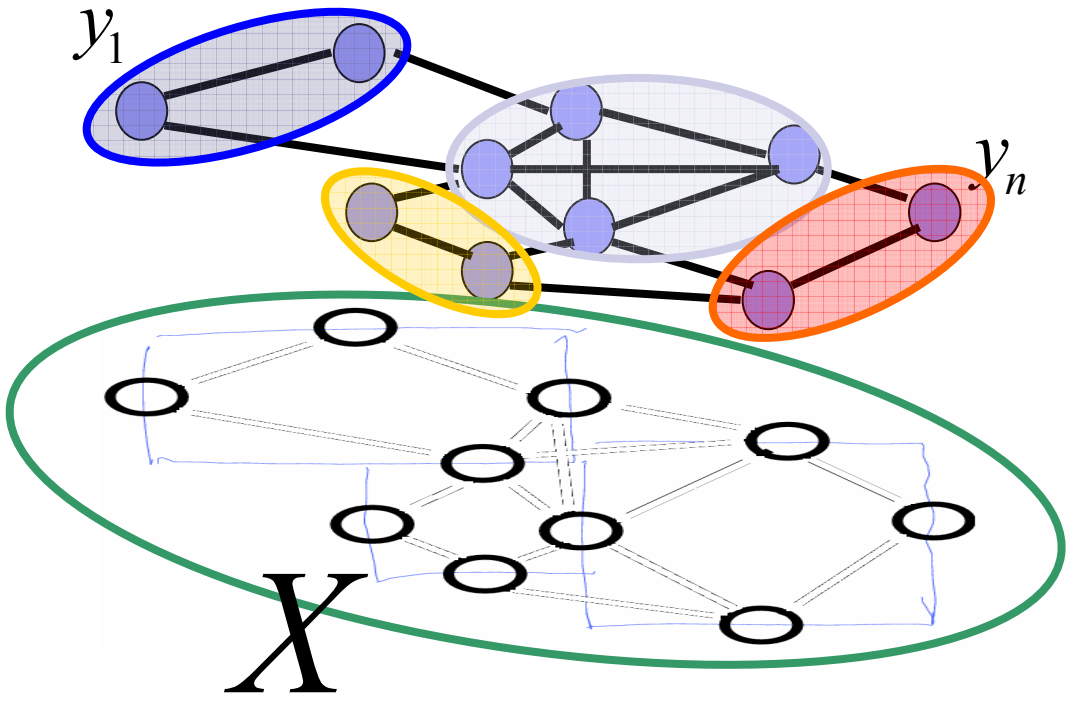
•Поскольку наблюдаемые переменные не входят в марковскую сеть, они не входят в клики

•Однако, за счет глобальной обусловленности по X, все потенциалы сети могут зависеть от данных

- Структура такой марковской сети задает функцию распределения:

$$p(Y \mid X) = \frac{1}{Z(X)} \prod_C \psi_C(Y_c, X)$$

Условное случайное поле



•Поскольку наблюдаемые переменные не входят в марковскую сеть, они не входят в клики

•Однако, за счет глобальной обусловленности по X, все потенциалы сети могут зависеть от данных

- Соответствующая функция энергии:

$$E(Y \mid X) = -\sum_C \ln \psi_C(Y_c, X) + \ln Z(X)$$

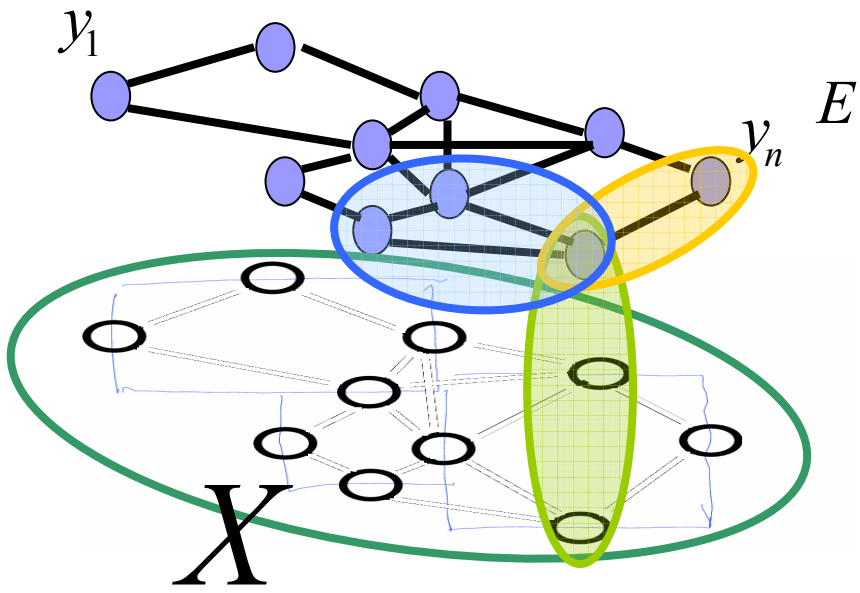
$$\arg\min_Y E(Y \mid X) = \arg\min_Y \left(-\sum_C \ln \psi_C(Y_c, X) \right)$$

Вопросы?

План лекции

- Алгоритмы обмена сообщениями
- Условное случайное поле
- Методы настройки потенциалов случайных полей

Настройка потенциалов поля



$$E(Y \mid X) = -\sum_C \ln \psi_C(Y_c, X) + \ln Z(X)$$

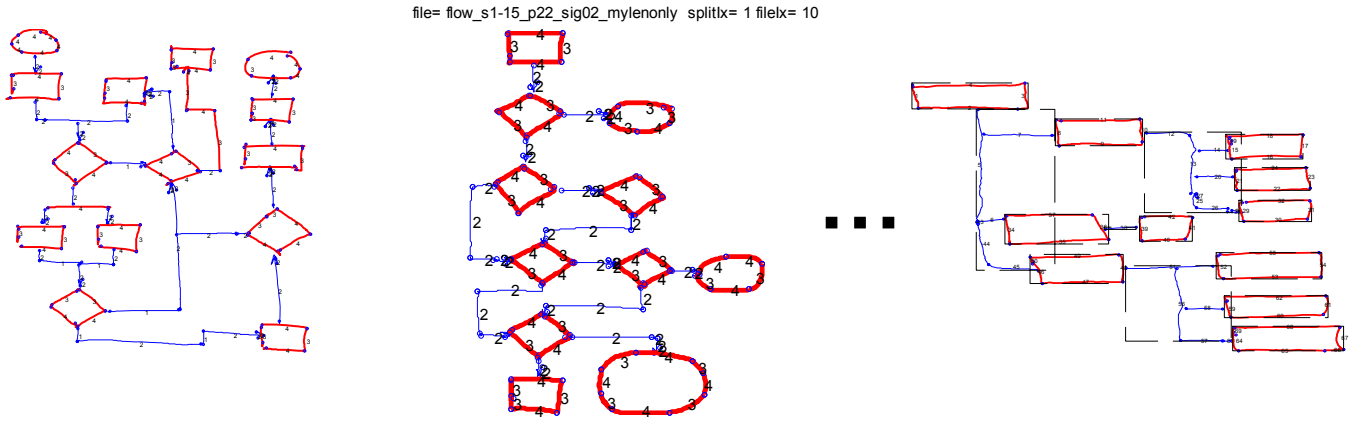
$$\psi_C(Y_C, X) = \exp\left(\sum_{k=1}^{K} w_k \varphi_k(Y_C, X)\right)$$

$$E(Y \mid X) = -\sum_C \sum_{k=1}^{K} w_k f_k(Y_C, X) + \ln Z(X, \mathbf{w}) = \ln Z(X, \mathbf{w}) - \mathbf{w}^T \Phi(Y, X)$$

Настройка потенциалов поля

•Если есть набор размеченных примеров, можно подобрать коэффициенты ***w*** так, чтобы на известных примерах минимизация энергии давала правильные ответы

$$(X_1, Y_1), \ldots (X_N, Y_N) \rightarrow \mathbf{w}^*$$



Настройка потенциалов поля

•Принцип максимального правдоподобия:

$$\mathbf{w}^* = \arg\max_{\mathbf{w}} \sum_{i=1}^{N} p(Y_i \mid X_i, \mathbf{w}) = \arg\min_{\mathbf{w}} \sum_{i=1}^{N} \left(\ln Z(X, \mathbf{w}) - \mathbf{w}^T \Phi(Y, X) \right)$$

•Принцип максимума апостериорной вероятности:

$$\mathbf{w}^* = \arg\max_{\mathbf{w}} \sum_{i=1}^{N} p(\mathbf{w}) p(Y_i \mid X_i, \mathbf{w}) =$$

$$= \arg\min_{\mathbf{w}} \sum_{i=1}^{N} \left(\ln Z(X, \mathbf{w}) - \mathbf{w}^T \Phi(Y, X) - \ln p(\mathbf{w}) \right)$$

Настройка потенциалов поля

- Легко посчитать первую и вторую производные относительно любой компоненты вектора ***w***
- Для настройки параметров модели используют численные методы оптимизации:
  - Метод сопряженных градиентов
  - L-BFGS

Недостатки

- Вычисление точных значений производных требует вычисления *Z(X)*
  - То есть перебора всех возможных вариантов разметки для каждой вершины
  - Эффективные алгоритмы вычисления *Z(X)* существуют только для графов простой структуры (цепочки, деревья)
- Время вычисления производных экспоненциально растет с увеличением количества переменных
- Приближенное вычисление производных ненадежно для обучения

Минимизация регуляризованного риска

•Принцип максимума апостериорной вероятности:

$$\mathbf{w}^* = \arg\max_{\mathbf{w}} \sum_{i=1}^{N} p(\mathbf{w})\, p(Y_i \mid X_i, \mathbf{w})$$

•Пример: поощряются маленькие значения параметров

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} \left(\|\mathbf{w}\|^2 + C \sum_{i=1}^{N} \left(\ln Z(X, \mathbf{w}) - \mathbf{w}^T \Phi(Y, X) \right) \right)$$

Частный случай функции потерь

Минимизация регуляризованного риска

Минимизация регуляризованного риска

•Пусть задана функция потерь для разметки $\Delta\left(Y, F(X)\right)$

- Пример: функция Хэмминга

$$\Delta\left(Y, F(X)\right) = \sum_{j=1}^{M} I\left[Y^{(j)} \neq F(X)^{(j)}\right]$$

•Требуется минимизировать мат. ожидание потерь

$$\int_{\mathbf{X}\times\mathbf{Y}} \Delta\left(Y, F(X)\right) dP\left(X, Y\right) \to \min_{F}$$

•Поскольку минимизировать мат. ожидание невозможно вычислить напрямую, будем минимизировать среднее значение функции потерь на размеченных данных

•Для получения устойчивого к шуму решения, используем регуляризацию

Минимизация регуляризованного риска

- Минимизация регуляризованного риска:

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} \left(\|\mathbf{w}\|^2 + C \sum_{i=1}^{N} \Delta\left(Y_i, F(X_i)\right) \right)$$

$$\mathbf{w} \longrightarrow E(X, Y, \mathbf{w}) \longrightarrow F(X_i) = \arg\min_{Y} E(X_i, Y, \mathbf{w}) \longrightarrow L(\mathbf{w}) = \sum_{i=1}^{N} \Delta\left(Y_i, F(X_i)\right)$$

$$F(X_i) = \arg\min_{Y} E(X_i, Y, \mathbf{w}) = \arg\min_{Y} \left(\ln Z(X, \mathbf{w}) - \mathbf{w}^T \Phi(Y, X) \right) =$$

$$= \arg\min_{Y} \left(-\mathbf{w}^T \Phi(Y, X) \right) = \arg\max_{Y} \left(\mathbf{w}^T \Phi(Y, X) \right)$$

Минимизация регуляризованного риска

•В простейшем случае найдется вектор ***w*** такой, что путем минимизации энергии с параметрами ***w*** можно получить правильные ответы на всех размеченных примерах, т. е

$$\forall i : \max_{Y \in \mathbf{Y} \setminus Y_i} \left\{ \mathbf{w}^T \Phi(X_i, Y) \right\} < \mathbf{w}^T \Phi(X_i, Y_i)$$

•Что эквивалентно условию

$$\forall i, \forall Y \in \mathbf{Y} \setminus Y_i : \mathbf{w}^T \left(\Phi(X_i, Y_i) - \Phi(X_i, Y) \right) > 0$$

Минимизация регуляризованного риска

•Поскольку более, чем один вектор параметров ***w*** удовлетворяет данному условию, введем дополнительное ограничение $\|\mathbf{w}\| \le 1$ и будем искать такой вектор параметров ***w***, при котором энергия ближайшего к оптимальному решения отличается от энергии оптимального решения не менее, чем на 1

•Получим задачу квадратичного программирования

$$SVM_0 : \quad \min_{\mathbf{w}} \tfrac{1}{2}\|\mathbf{w}\|^2$$

$$\forall i, \forall Y \in \mathbf{Y} \setminus Y_i : \mathbf{w}^T \Phi(X_i, Y_i) - \Phi(X_i, Y) \ge 1$$

•Задача включает огромное число неравенств, однако система неравенств избыточна

Минимизация регуляризованного риска

Алгоритм

Пока веса ***w*** не меняются (более чем на константу)

Цикл по всем размеченным примерам:

1. Вычислить наиболее вероятную разметку примера n

$$Y^* = \arg\max_Y \left(\mathbf{w}^T \Phi(Y, X_n) \right)$$

2. Если $Y^* \neq Y_n$, добавить Y^* в список отрицательных примеров

$$\mathbf{S} = \mathbf{S} \cup \left\{ Y^* \right\}$$

3. Обновить параметры так, чтобы правильная разметка имела наименьшую энергию

$$SVM_0 : \quad \min_{\mathbf{w}} \tfrac{1}{2} \|\mathbf{w}\|^2$$

$$\forall i, \forall Y \in \mathbf{S}_i : \mathbf{w}^T \left(\Phi(X_i, Y_i) - \Phi(X_i, Y) \right) \geq 1$$

Минимизация регуляризованного риска

•Не всегда можно найти такой вектор параметров ***w***, чтобы минимизация энергии на всех размеченных примерах давала верный результат

•Ослабим требования, за счет введения дополнительных переменных

$$SVM_1: \quad \min_{\mathbf{w}} \tfrac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{n}\sum_{i=1}^{n}\xi_i, \xi_i \geq 0$$

$$\forall i, \forall Y \in \mathbf{Y} \setminus Y_i : \mathbf{w}^T\left(\Phi(X_i, Y_i) - \Phi(X_i, Y)\right) \geq 1 - \xi_i$$

•Для того, чтобы учесть функцию потерь изменим формулировку задачи

$$SVM_1^{\Delta_s}: \quad \min_{\mathbf{w}} \tfrac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{n}\sum_{i=1}^{n}\xi_i, \xi_i \geq 0$$

$$\forall i, \forall Y \in \mathbf{Y} \setminus Y_i : \mathbf{w}^T\left(\Phi(X_i, Y_i) - \Phi(X_i, Y)\right) \geq 1 - \frac{\xi_i}{\Delta\left(Y, Y_i\right)}$$

Пример

•Бинарная сегментация изображения на область, соответствующую объекту (корове), и область, соответствующую фону:

$$E\left(X_n, Y_n; \mathbf{w}\right) = w_1 \sum_{i \in V} -\ln P\left(X_n^{(i)} \mid Y_n^{(i)}\right) + w_2 \sum_{i,j \in N} I\left[Y_n^{(i)} \neq Y_n^{(j)}\right] +$$

$$+ w_2 \sum_{i,j \in N} I\left[Y_n^{(i)} \neq Y_n^{(j)}\right] C\left(X_n^{(i)}, X_n^{(j)}\right)$$

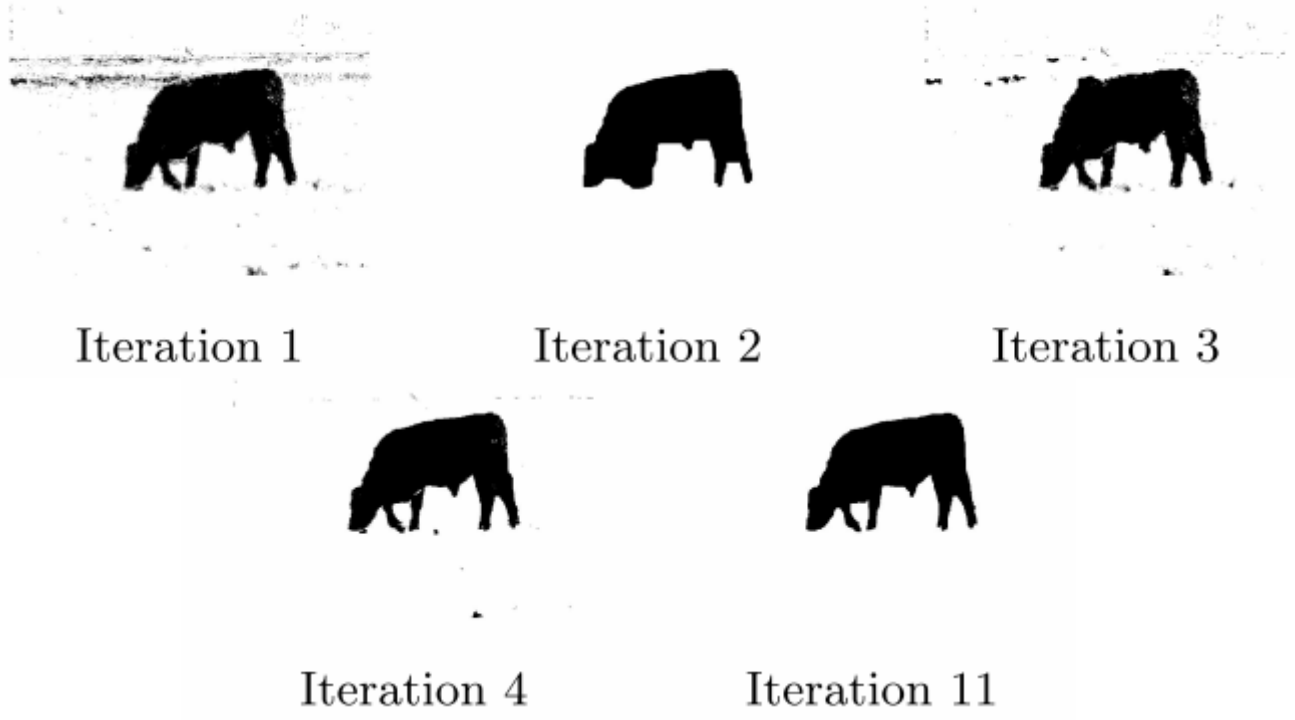
Цветовая модель объекта и фона

Исходное изображение

Желаемая разметка

Пример

- Метод сходится за 11 итераций.
- Решение на каждой итерации добавляется в множество ограничений

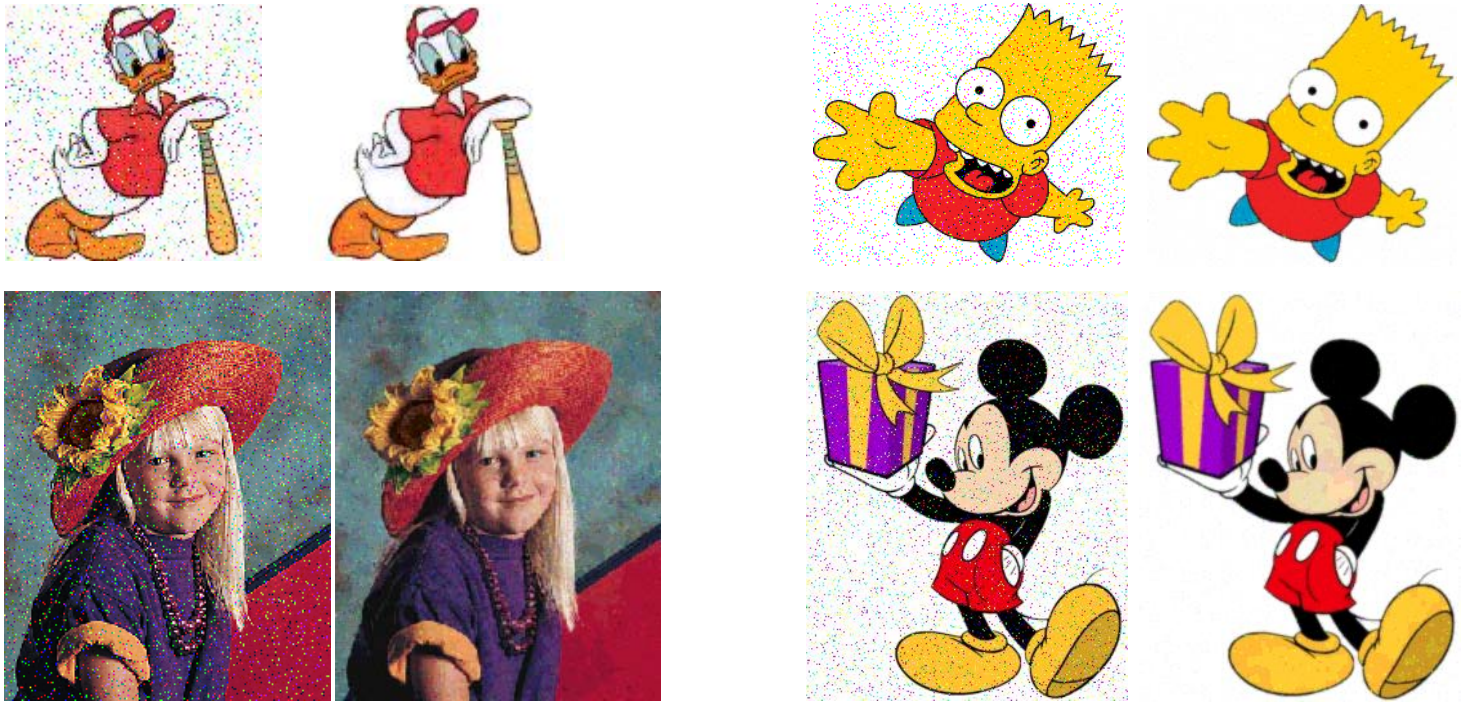


Вопросы?

Список литературы

- Christopher M. Bishop, **Pattern Recognition and Machine Learning,** 2006
- Hanna M. Wallach. **Conditional Random Fields: An Introduction**, 2004.
- Ioannis Tsochantaridis, Thomas Hofmann, Thorsten Joachims, Yasemin Altun. **Support Vector Machine Learning for Interdependent and Structured Output Spaces**, ICML, 2004.
- Martin Szummer, Pushmeet Kohli, and Derek Hoiem, **Learning CRFs using Graph Cuts**, ECCV 2008

Первое задание: восстановление изображений



http://courses.graphicon.ru/main/smisa2009