Байесовские сети доверия II

Сергей Николенко

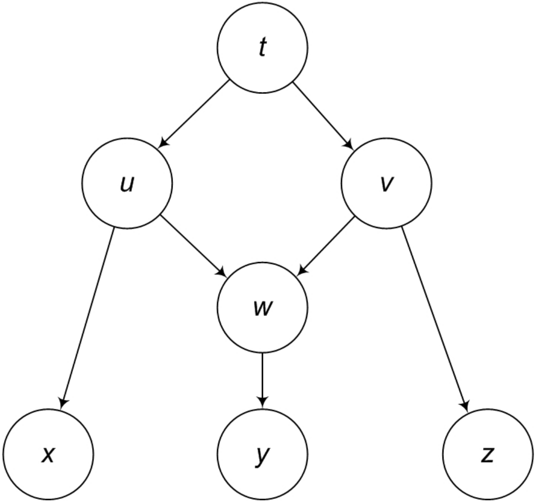
Машинное обучение — ИТМО, осень 2006

Outline

Мотивация

- На предыдущей лекции мы научились обрабатывать БСД в виде полидеревьев.
- Как обрабатывать циклы? Наш алгоритм не будет работать: в одну и ту же вершину из другой можно будет придти несколькими путями.
- На этой лекции мы разберём другой алгоритм, который работает в этом более общем случае.

Наш пример



Доменный граф

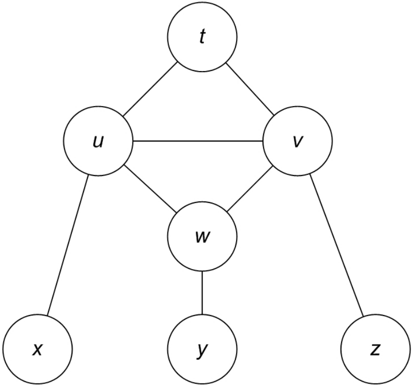
Определение

Пусть $\mathcal{P} = \{p(\tilde{x}_1|\tilde{X}_1), \ldots, p(\tilde{x}_m|\tilde{X}_m)\}$ *— набор распределений вероятностей над множеством атомарных событий* $\mathcal{S} = \{x_1, \ldots, x_n\}$*. Доменный граф для* $\mathcal{P}$ *— это ненаправленный граф, вершины которого — элементы* $\mathcal{S}$*; две вершины связаны ребром тогда и только тогда, когда они обе встречаются в одном и том же распределении вероятностей* $p(\tilde{x}_i|\tilde{X}_i) \in \mathcal{P}$*.*

Доменный граф

- В случае БСД это всего лишь означает, что граф теперь ненаправленный, и в нём добавляются рёбра между родителями общего предка.
- Такие рёбра называются *моральными*, а сам доменный граф в случае БСД — моральным графом.

Моральный граф

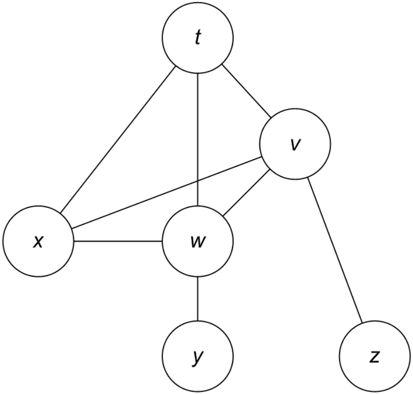


Добавилось ребро между u и v — общими предками w.

Дальнейший план

- Мы хотим в дальнейшем по одной удалять вершины из морального графа, при этом проецируя общее распределение на то, что останется.
- Каждый раз, когда мы удаляем вершину, нам приходится объединять её соседей, потому что нужно объединить распределение вероятностей.

Пример



Вот результат элиминации вершины u.

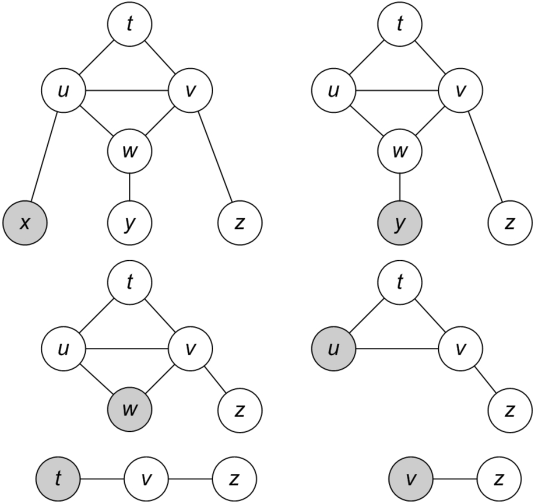
Идеальная элиминирующая последовательность

Цель понятна — нужно постараться избегать добавления новых ребер при элиминировании переменных.

Определение

Идеальной элиминирующей последовательностью называется такая последовательность переменных, что при их элиминировании в этой последовательности ни разу ни одного ребра в моральный граф добавлено не будет.

Идеальная элиминирующая последовательность: пример



Симплициальные вершины

- Если $x_1, \ldots, x_i, \ldots, x_l$ — идеальная элиминирующая последовательность, и множество соседей x_i образует полный подграф, то $x_i, x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_l$ — также идеальная элиминирующая последовательность. Такие вершины называются *симплициальными*.
- Рассмотрим множество клик, которые появятся в моральном графе в процессе последовательной элиминации переменных по какой-либо последовательности. Выбросим подграфы других клик. Это множество всегда есть множество клик морального графа сети.

Упражнение

Докажите эти два утверждения.

Триангулярные графы

Определение

Ненаправленный граф называется триангулярным, если у него существует идеальная элиминирующая последовательность.

Это называется триангулярностью, потому что эквивалентно тому, что любой цикл длиной больше 3 разбит на меньшие (две его несоседние вершины соединены).

Триангулярные графы и симплициальные вершины

Теорема

У неполного триангулярного графа, содержащего по крайней мере три вершины, всегда есть две несоседних симплициальных вершины.

Следствие: Ненаправленный граф триангулярен тогда и только тогда, когда все его вершины можно элиминировать, последовательно элиминируя симплициальные вершины.

Триангуляция

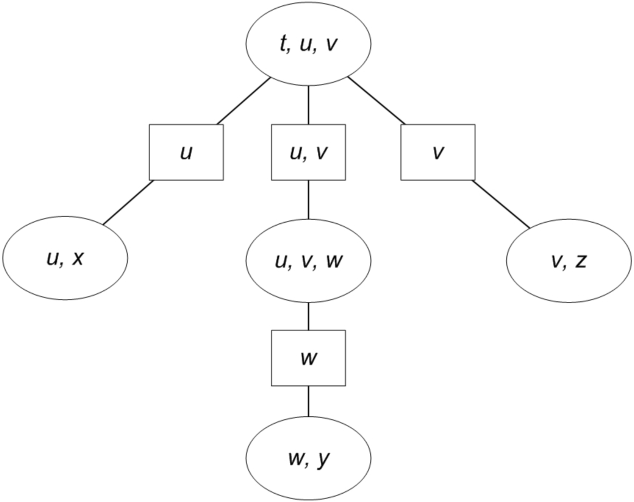
- Мы хотим действовать по идеальной элиминирующей последовательности.
- Чтобы она существовала, нужно, чтобы граф был триангулирован.
- Но это не обязательно так; значит, надо триангулировать.
- Об этом чуть позже.

Дерево смежности

Определение

Дерево смежности (join tree) — это дерево, вершинами которого служат элементы $\mathrm{Clique}(G)$ *(клики графа* G*), а ребра организованы так, что для любых двух вершин дерева смежности* C_1, C_2 *любая вершина на пути от* C_1 *к* C_2 *содержит их пересечение* $C_1 \cap C_2$*.*

Дерево смежности: пример



Дерево смежности и триангуляция

Теорема

Ненаправленный граф триангулярен тогда и только тогда, когда для него существует дерево смежности.

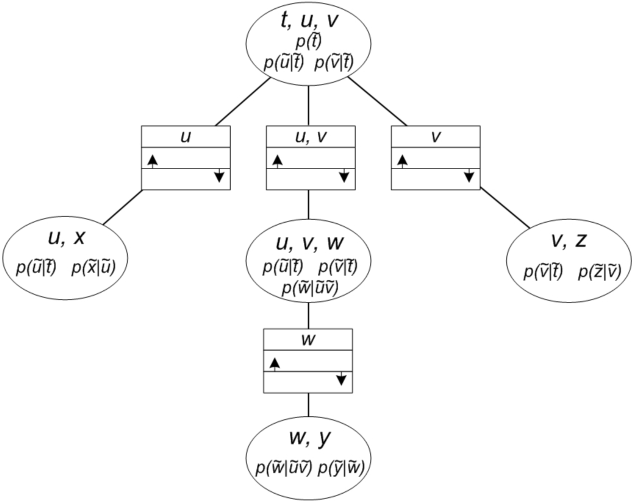
Дерево сочленений

Definition

Дерево сочленений (junction tree) морального графа — это его дерево смежности, где каждой вершине дополнительно приписаны условные вероятности исходной БСД, области определения которых полностью содержатся в соответствующей клике морального графа, а каждый из сепараторов содержит два *почтовых ящика*, предназначенных для передачи пересчитанных тензоров условных вероятностей в одну и другую сторону.

Конечно, реально дерево смежности и дерево сочленений не отличаются.

Дерево сочленений: пример



Дерево сочленений

- Дерево сочленений — основной объект, на котором будет действовать алгоритм.
- Как его построить?
- Для этого нужно построить *идеальную нумерацию*.

Идеальная нумерация

Определение

Фиксируем граф $G = (V, E)$. *Нумерация*

$$\sigma : \{1, \ldots, |V|\} \longrightarrow V$$

называется идеальной, если для всякого i *множество вершин*

$$\mathrm{Fam}(\sigma(i)) \cap \sigma(\{1, \ldots, i\})$$

индуцирует полный подграф G.

Алгоритм построения идеальной нумерации

- Для всех i от 1 до $|V|$:
  - найти еще не пронумерованную вершину x, для которой $|\mathrm{Fam}(x) \cap \sigma(1, \ldots, i-1)|$ максимальна (если таких вершин несколько, выбрать любую произвольным образом);
  - присвоить ей номер i, т.е. положить $\sigma(i) = x$.
- Выдать полученную нумерацию.

Алгоритм триангуляции

Простейший алгоритм триангуляции — запустить алгоритм, получить «идеальную» нумерацию, а потом сделать её идеальной:

- Построить нумерацию σ при помощи предыдущего алгоритма.
- Для всех i от $|V|$ до 1:

$$E = E \cup \{(x, y) \mid \\ \mid x, y \in \mathrm{Fam}(\sigma(i)) \cap \sigma(\{1, \ldots, i-1\}, (x, y) \notin E\}.$$

- Выдать граф (V, E).

Алгоритм построения дерева смежности

- $\boldsymbol{C} = \emptyset$.
- $\boldsymbol{E} = \emptyset$.
- Для всех i от 1 до $|V|$:
    - найти еще не пронумерованную вершину x, для которой $|\mathrm{Fam}(x) \cap \sigma(1, \ldots, i-1)|$ максимальна (если таких вершин несколько, выбрать любую произвольным образом);
    - присвоить ей номер i, т.е. положить $\sigma(i) = x$;
    - $C_i = \{\mathrm{Fam}(\sigma(i)) \cap \sigma(\{1, \ldots, i\})\}$;
    - $\boldsymbol{C} = \boldsymbol{C} \cup \{C_i\}$;
    - $\boldsymbol{E} = \boldsymbol{E} \cup (C_i, C_j)$, где $j = \max_{k<i,(i,k)\in E} k$.
- Для всех i от 1 до $|V|$, если C_i полностью содержится в одном из своих соседей ($\exists j: \ C_i \subseteq C_j \wedge (C_i, C_j) \in \boldsymbol{E}$), то удалить C_i из $\boldsymbol{C}$ и соединить соседей C_i напрямую.
- Выдать граф $(\boldsymbol{C}, \boldsymbol{E})$.

Outline

Схема алгоритма

Итак, вот что у нас пока есть:

- Морализовать базовый граф БСД.
- Триангулировать моральный граф.
- Построить дерево смежности (оно же дерево сочленений).
- Выполнять собственно пропагацию.

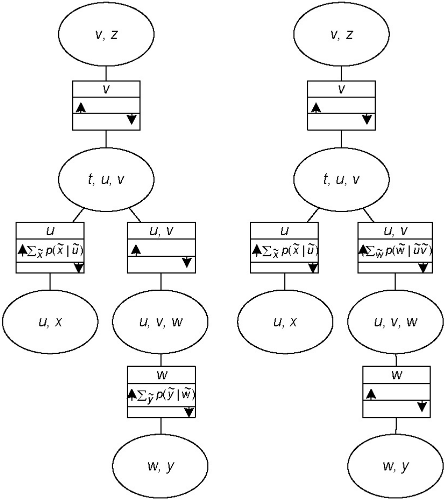
Первые три шага мы уже рассмотрели, дело за четвёртым.

Сбор сведений

Это первый этап алгоритма. Предположим, что мы хотим найти z.

- Сначала нужно найти вершину дерева сочленений, содержащую z. В данном случае такая клика одна — $\{v, z\}$.
- Эта клика становится временным корнем дерева, а остальные вершины, начиная с листьев, посылают к временному корню сообщения, в которых записан результат суммирования по переменным, не содержащимся в ближайшем сепараторе.
- Каждая вершина посылает сообщение «наверх» тогда, когда она получила все сообщения «снизу».

Первые два шага



Условие передачи сообщения

Весь алгоритм можно представить как последовательную передачу сообщений между узлами тогда, когда выполняется условие:

Определение

Пусть в моральном графе есть клика C, которой соответствует набор распределений вероятностей Ψ_C, а у нее — соседний сепаратор S. Пусть остальные соседи C — $S_1, \ldots, S_k$ — уже получили сообщения Ψ_i от своих других соседей, т.е. во входящих почтовых ящиках $S_1, \ldots, S_k$ уже лежат готовые распределения вероятностей. Назовем такую ситуацию условием передачи сообщения.

Алгоритм

- На этапе сбора сведений условие выполняется снизу вверх, и узлы передают сообщения наверх.
- Потом, когда мы добрались до корня и пересчитали там, мы начинаем двигаться обратно; условие не меняется, но теперь оно работает для всех исходящих сепараторов.
- Алгоритм заканчивает работу, когда пустых почтовых ящиков больше нет.

Алгоритм

Алгоритму абсолютно всё равно, есть ли в сети какие-то означивания или нет. Если они есть, значения вероятностей изменятся, и только. Структура алгоритма останется прежней.

Упражнения

Упражнение

Реализовать описанный в этой лекции алгоритм пропагации.

Спасибо за внимание!

- Lecture notes, слайды и коды программ появятся на моей homepage:
  `http://logic.pdmi.ras.ru/~sergey/index.php?page=teaching`
- Присылайте любые замечания, коды программ на других языках, решения упражнений, новые численные примеры и прочее по адресам:
  `sergey@logic.pdmi.ras.ru`, `smartnik@inbox.ru`