Синтез множества минимальных графов смежности: статистическая оценка сложности инкрементального алгоритма[1]

Березин А. И., студент мат.-мех. фак-та кафедры информатики СПбГУ, стажер лаб. ТиМПИ СПИИРАН, beraliv.spb@gmail.com

Иванова А. В., студент мат.-мех. фак-та кафедры информатики СПбГУ, s.tigma@yandex.ru

Зотов М. А., студент мат.-мех. фак-та кафедры информатики СПбГУ, стажер лаб. ТиМПИ СПИИРАН, zotov1994@mail.ru

Аннотация

В настоящей работе представлен инкрементальный алгоритм генерации множества минимальных графов смежности, проведены статистические тесты, показывающие прирост в скорости работы данного алгоритма над прямым алгоритмом генерации множества минимальных графов смежности. Статистический анализ производился на диапазоне от 4 до 12 вершин и ограничивается этим числом ввиду экспоненциального роста сложности работы алгоритмов. Для удобства восприятия результаты экспериментов приведены на графиках.

Введение

В теории алгебраических байесовских сетей в качестве вторичной структуры могут выступать так называемые графы смежности [7, 11] и, как частный случай, — минимальные графы смежности (МГС). Такие графы хранят в себе данные о предметной области — фрагменты знаний (ФЗ) [10], их связи и зависимости; кроме того, такие графы могут быть визуализированы.

В качестве алгоритмов синтеза МГС были разработаны прямой, жадный, инкрементальный и декрементальный алгоритмы [1, 3, 6, 11].

[1]Статья содержит материалы исследований, частично поддержанных грантом РФФИ 15-01-09001 — «Комбинированный логико-вероятностный графический подход к представлению и обработке систем знаний с неопределенностью: алгебраические байесовские сети и родственные модели».

Последние два достраивают имеющийся МГС до нового МГС при добавлении и удалении одной вершины соответственно. Первые два алгоритма, напротив, осуществляют синтез МГС “с нуля”, т.е. не используют имеющуюся структуру АБС в качестве базы или основания для добавления вновь прибывшего ФЗ. Прямой и жадный алгоритмы часто используются для создания такого основания, т.е. тогда, когда вторичная структура над ФЗ ещё не была синтезирована.

Статистический анализ сложностей указанных алгоритмов был осуществлен и проанализирован в статьях [1, 2, 3, 4, 5, 6]. В частности, было установлено, что инкрементальный алгоритм имеет значительное скоростное преимущество перед прямым и жадным алгоритмами на мощностях графа 10–100.

С целью выявления особенностей и скрытых свойств МГС, актуальной задачей является синтез множества всех МГС по данному набору ФЗ. Схема генерации этого множества, а также теоретические оценки сложности были приведены в [12, 13]. Однако имплементация схемы, равно как и разработка и анализ конкурирующих аппаратов синтеза всевозможных минимальных графов смежности, — проведены не были. Таким образом, целью настоящей работы является, с одной стороны, разработка и реализация алгоритмов синтеза множества всех МГС в ситуации постоянного изменения первичной структуры, с другой стороны, сравнительный анализ и проведение вычислительных экспериментов над конкурирующими алгоритмами такого синтеза.

Определения и обозначения

Воспользуемся системой терминов и обозначений, сложившихся в [7, 8, 9]. Пусть задан конечный алфавит символов A, а непустые множества символов (без повторов) — слова — рассматриваются как возможные значения нагрузок вершин графов и их ребер. Пусть имеется набор вершин $V = \{v_1, \ldots, v_n\}$ и нагрузки $W = \{w_1, \ldots, w_n\}$, причем W_u является нагрузкой для вершины u. Мощностью графа G будем называть число вершин в этом графе.

Назовем неориентированный граф $G = \langle V, E \rangle$ графом смежности, если он удовлетворяет следующим условиям:

1. $\forall u, v \in V : \exists$ путь P в графе $G : \forall s \in P \Rightarrow W_e = W_u \cap W_v \neq \varnothing$,
2. $\forall e = \{u, v\} \in E \Rightarrow W_e = W_u \cap W_v \neq \varnothing$,

3. $\nexists u, v \in V : W_u \subseteq W_v$, — нагрузка одной вершины графа не входит полностью в нагрузку любой другой вершины.

Граф смежности с минимальным (максимальным) числом ребер мы будем называть минимальным (максимальным) графом смежности G_{min} (G_{max}). Максимальный граф смежности всегда единственен [9].

Сепаратором двух вершин называется пересечение нагрузок этих вершин.

Сужением — $G \downarrow s$ — назовем $\{\{v|v \in V, s \subseteq W_v\}, \{e|e \in E, s \subseteq W_e\}\}$. Под сужением на сепаратор s без указания конкретного графа сужения будем понимать такое сужение: $G_{max} \downarrow s$.

Сильное сужение — $G \downarrow s$ — обозначим как $\{\{v|v \in V, s \subseteq W_v\}, \{e|e \in E, s \subseteq W_e\}\}$. Под сильным сужением на сепаратор s без указания конкретного графа сужения будем понимать такое сильное сужение: $G_{max} \downarrow s$.

Владениями назовем компоненты связности сильного сужения на данный сепаратор.

Сепараторы классифицируем следующим образом:

1. Бисепаратор — сепаратор ($s \in \mathrm{BSep}$), который образует обязательное ребро, т.е. ребро, которое будет встречаться в каждом МГС [9].

2. Стереосепаратор — сепаратор ($s \in \mathrm{SSep}$), у которого как минимум два владения [9].

МК-пара обозначает пару {NecessaryEdges, StereoHoldings}, которая была построена по первичной структуре Workloads — множеству нагрузок вершин.

1. $\mathrm{NecessaryEdges} = \{\{s,\ E(\downarrow s)\} | s \in \mathrm{BSep}\}$ — словарь, который возвращает обязательное ребро по соответствующему бисепаратору.

2. $\mathrm{StereoHoldings} = \{\{s, \mathrm{Components}(E(\downarrow s))\} | s \in \mathrm{SSep}\}$ — словарь, возвращающий владения по заданному стереосепаратору.

Инкрементальной МК-парой мы называем такую МК-пару, в которой хранятся все бисепараторы и стереосепараторы, которые изменились при удалении нагрузки вершины из начального множества Workloads.

Алгоритмом сборки назовем любой алгоритм, генерирующий подмножество МГС по МК-паре. Под инкрементальным алгоритмом сборки следует понимать алгоритм, синтезирующий подмножество МГС по инкрементальной МК-паре.

Инкрементальный алгоритм синтеза множества минимальных графов смежности

Алгоритм инкрементального синтеза множества минимальных графов смежности состоит из двух этапов: формирование инкрементальной МК-пары и сборка такой пары. Указанный алгоритм был разбит на два алгоритма (Листинги 1 и 2 соответственно). Рассмотрим каждую из частей подробнее.

В первой части обрабатывается всё множество сепараторов, которое образуется при добавлении новой вершины с нагрузкой. Суть данной обработки заключается в выявлении бисепараторов и стереосепараторов. Указанная обработка необходима для формирования инкрементальной МК-пары.

Во второй части происходит инкрементальная сборка структуры, сформированной на первом этапе. После окончания работы алгоритм возвращает множество минимальных графов смежности. Код алгоритма синтеза множества всевозможных графов смежности приведен в Листинге 3.

На первом этапе необходимо обработать всё множество сепараторов, которое образуется при добавлении новой вершины с нагрузкой: необходимо выявить бисепараторы и стереосепараторы. Указанная обработка необходима для формирования инкрементальной МК-пары. Формирование указанного множества реализовано с помощью функции GetIncSeparatorSet (Листинг 1). Дополнительно отметим, что элементы такого множества могут пересекаться с элементами множества сепараторов, которое было построено до внесения нагрузки вершины: такие сепараторы также подлежат рассмотрению. Непосредственно обработка сепаратора заключается в проверке, является ли он бисепаратором или стереосепаратором. Полученные результаты сохраняются в инкрементальной МК-паре, для построения которой введена функция GetIncMK1 (Листинг 2).

Алгоритм принимает на вход множество сепараторов SeparatorSet, построенного по первичной структуре Workloads, а также добавленная вершина w'. Сначала алгоритм сохраняет предыдущую таблицу сепараторов (строка 1). Затем перебираются все нагрузки вершин (строка 2) и сохраняются сепараторы старой вершины и только что добавленной (строка 3). Если сепаратор непустой, то он добавляется в множество сепараторов (строки 4–5).

Алгоритм принимает на вход множество сепараторов SeparatorSet, построенного по первичной структуре Workloads, добавленную верши-

Листинг 1 Алгоритм построения множества непустых сепараторов GetIncSeparatorSet

Input: SeparatorSet = ⟨Separators, SeparatorTable⟩, Workloads, w'
Output: ⟨Separators′, SeparatorTable′⟩

1: SeparatorTable′ = SeparatorTable
2: **foreach** (w **in** Workloads)
3: SeparatorTable′$[w, w'] = w \cap w'$
4: **if** ($w \cap w' \neq \varnothing$)
5: Separators′ = Separators′ ∪ SeparatorTable′$[w, w']$

Листинг 2 Алгоритм GetIncMK1 синтеза инкрементальной МК-пары

Input: SeparatorSet = ⟨Separators, SeparatorTable⟩, ⟨NecessaryEdges, StereoHoldings⟩, Workloads, w'
Output: IncMK

1: ⟨Separators′, SeparatorTable′⟩ =
2: **GetIncSeparatorSet**(SeparatorSet, Workloads)
3: **foreach** (s **in** Separators′)
4: Vertices = **NarrowedVertices**(s, Workloads ∪ $\{w'\}$)
5: **if** (|Vertices| = 2)
6: NecessaryEdges′[s] = ⟨Vertices[0], Vertices[1]⟩
7: **else**
8: Holdings = **Components**(**StrongNarrowing**(Vertices, s, SeparatorTable′))
9: **if** |Holdings| > 1 **then**
10: StereoHoldings′[s] = Holdings

ну с указанной нагрузкой w′ и МК-пару, построенной до внесения нового ФЗ. Сперва, строится множество множество сепараторов (строки 1–2), используя алгоритм GetIncSeparatorSet (Листинг 1). Затем рассматривается каждый сепаратор и обрабатываются только рассмотренные нами разновидности (бисепаратор — строки 5–6, стереосепаратор — строки 7–10).

Алгоритм принимает на вход обычную и инкрементальную МК-пары, первичную структуру, добавленную нагрузку w′, множество жил SinewSet, построенных по первичной структуре и тип построения жил T.

Во втором этапе происходит инкрементальная сборка построенной инкрементальной МК-пары. Сначала проверяются все обязательные ребра, которые были построены до добавления нагрузки в первичную структуру (строки 1–3). После этого перебираются стереосепараторы.

Листинг 3 Инкрементальный алгоритм сборки IncAssembly

Input: ⟨NecessaryEdges, StereoHoldings⟩, ⟨NecessaryEdges′, StereoHoldings′⟩, Workloads, w', SinewSet, T
Output: SMJG

```
1:  foreach (⟨s, e⟩ in NecessaryEdges)
2:      if (s ⊄ w′)
3:          NecessaryEdges′[s] = e
4:  foreach (⟨s, h⟩ in StereoHoldings)
5:      if s ∉ StereoHoldings′
6:          if s ⊄ w′
7:              StereoHoldings′[s] = h
8:              SinewSet′[s] = SinewSet[s]
9:      else
10:         SinewSet′[s] = GetSinewSet(h, T)
11: foreach (⟨s, h⟩ in StereoHoldings′ \ StereoHoldings)
12:     SinewSet′[s] = SinewSet(h, T)
13: edgeSets = GetUAF(SinewSet′)
14: SMJG = ∅
15: foreach (edgeSet in edgeSets)
16:     SMJG = SMJG ∪ ⟨Workloads ∪ w′, NecessaryEdges′ ∪ edgeSet⟩
17: if (edgeSets = ∅)
18:     SMJG = ⟨Workloads ∪ w′, NecessaryEdges′⟩
```

Если стереовладения не расширились (строка 5) и не были еще рассмотрены в алгоритме GetIncMK1 (строка 6), то нужно сохранить предыдущие результаты. Иначе, если расширились, надо пересчитать жилы (строки 9–12). Затем действия повторяются как и для обычного алгоритма сборки (строки 13–18) [9].

Введем функцию GetSMJG для построения множества, описанного выше.

Листинг 4 Построение множества МГС GetSMJG1

Input: SeparatorSet = ⟨Separators, SeparatorTable⟩, MK = ⟨NecessaryEdges, StereoHoldings⟩,Workloads, w', SinewSet, T
Output: SMJG

```
IncMK = GetIncMK1(SeparatorSet, MK, Workloads, w′)
IncSMJG = IncAssembly(MK, IncMK, Workloads, w′, SinewSet, T)
```

Эмпирическая оценка относительных сложностей алгоритмов

Подробное описание схемы проведения вычислительных экспериментов, в основе которых лежит сравнительный статистический анализ, а также экспериментов, проведенных по указанной схеме, можно найти в [3, 5]. Именно поэтому ограничимся лишь кратким описанием вычислительных экспериментов.

Результаты экспериментов по сравнению скоростей работы прямого и инкрементального алгоритмов синтеза множества минимальных графов смежности представлены на Рис. 1–3.

Каждый график отображает результаты выходных данных алгоритма Experiment из [1]. В качестве статистик были выбраны среднее геометрическое — Rg, первый и девятый децили — D1 и D9 соответственно, а также минимумы и максимумы — MIN и MAX соответственно. На горизонтальной оси указано мощность первичной структуры, на вертикальной — отношение скоростей работы инкрементального алгоритма к прямому.

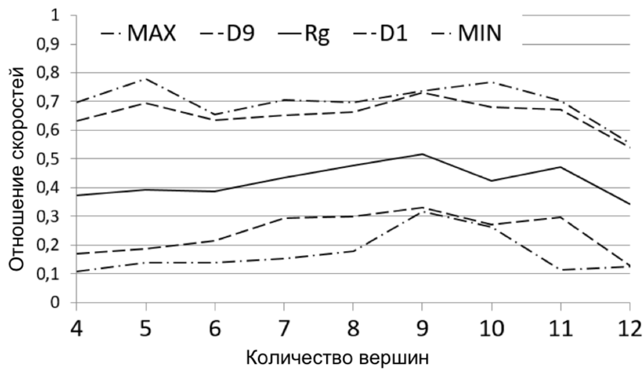

Рис. 1: Прямой и инкрементальный алгоритмы. Алфавит — 61 символов. 80% — 2–4 порядков, 17% — 5–7 порядков, 3% — 8–10 порядков, 0% — 11–13 порядков.

Анализ рисунков позволяет сделать вывод о том, что инкрементальный алгоритм превосходит в скорости работы прямой алгоритм на диапазоне вершин 4–12. Отметим, что по причине экспоненциального роста времени работы, при увеличении количества вершин в графе не представляется возможным проводить дальнейшие эксперименты.

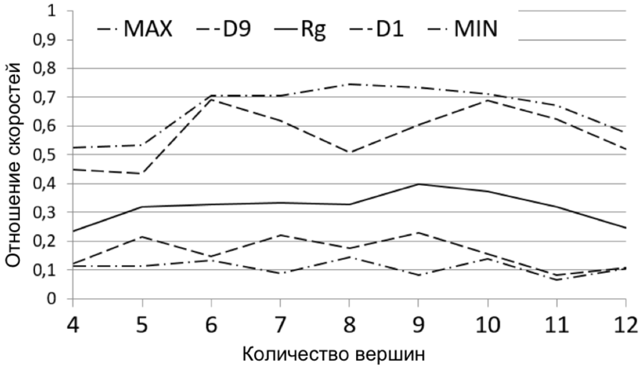

Рис. 2: Прямой и инкрементальный алгоритмы. Алфавит — 61 символов. 70% — 2–4 порядков, 17% — 5–7 порядков, 8% — 8–10 порядков, 5% — 11–13 порядков.

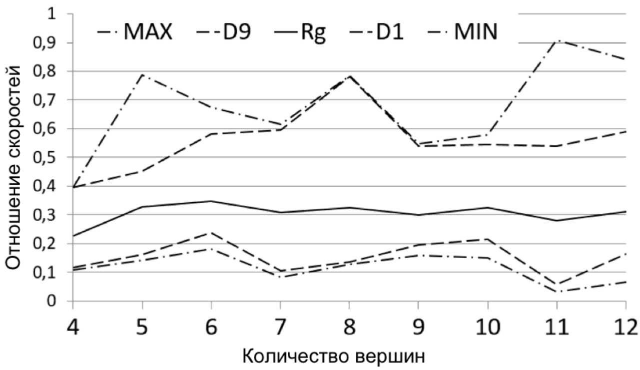

Рис. 3: Прямой и инкрементальный алгоритмы. Алфавит — 61 символов. 60% — 2–4 порядков, 20% — 5–7 порядков, 12% — 8–10 порядков, 8% — 11–13 порядков.

На приведенных рисунках нет признаков монотонного увеличения или уменьшения относительных времен работ алгоритмов, поэтому судить о преимуществе прямого или инкрементального алгоритмов на других диапазонах нельзя. Однако нельзя не отметить некоторую тенденцию на поддиапазоне 9–12 вершин на рисунках 1–2: инкрементальный алгоритм становится быстрее прямого.

Заключение

Разработан и реализован алгоритм инкрементального синтеза множества всех минимальных графов смежности, а также приведен его псевдокод; проведена серия экспериментов по сравнению данного алгоритма с конкурирующим алгоритмом — прямым. Статистические эксперименты показали, что прямой алгоритм уступает инкрементальному в скорости работы на заданном диапазоне вершин. Представленные результаты являются очередным шагом для дальнейших исследований множества вторичных структур в теории алгебраических байесовских сетей.

Литература

[1] Зотов М. А., Левенец Д. Г., Тулупьев А. Л., Золотин А. А. Синтез вторичной структуры алгебраических байесовских сетей: инкрементальный алгоритм и статистическая оценка его сложности // Научно-технический вестник информационных технологий, механики и оптики. 2016. Т. 16. № 1. С. 122–132.

[2] Зотов М. А., Тулупьев А. Л. Вторичная структура алгебраических байесовских сетей: статистическая оценка сложности прямого алгоритма синтеза // SCM'2015: XVIII Международная конференция по мягким вычислениям и измерениям. Санкт-Петербург, 19–21 мая 2015 г. С. 158–162.

[3] Зотов М. А., Тулупьев А. Л. Синтез вторичной структуры алгебраических байесовских сетей: методика статистической оценки сложности и компаративный анализ прямого и жадного алгоритмов // Компьютерные инструменты в образовании, 2015. № 1. С. 3–16

[4] Зотов М. А., Тулупьев А. Л. Синтез вторичной структуры алгебраических байесовских сетей: сравнительный анализ статистических оценок сложности двух алгоритмов. // VIII международная научно-практическая конференция «Интегрированные модели и мягкие вычисления в искусственном интеллекте». Коломна, 2015 г. Т. 2. С. 790–798.

[5] Зотов М. А., Тулупьев А. Л., Сироткин А. Л. Статистические оценки сложности прямого и жадного алгоритмов синтеза вторичной

структуры алгебраических байесовских сетей // Нечеткие системы и мягкие вычисления. 2015. Т. 10. №1. С. 75–91.

[6] Левенец Д. Г., Зотов М. А., Тулупьев А. Л. Инкрементальный алгоритм синтеза минимального графа смежности // Компьютерные инструменты в образовании, 2015. № 6. С. 3–18.

[7] Опарин В. В., Тулупьев А. Л. Синтез графа смежности с минимальным числом ребер: формализация алгоритма и анализ его корректности. // Тр. СПИИРАН. 2009. 11. С. 142–157.

[8] Тулупьев А. Л. Алгебраические байесовские сети: глобальный логико-вероятностный вывод в деревьях смежности: Учеб. пособие. СПб.: СПбГУ; ООО Издательство «Анатолия», 2007. С. 40. (Сер. Элементы мягких вычислений).

[9] Фильченков А. А. Синтез графов смежности в машинном обучении глобальных структур алгебраических байесовских сетей // Дисс. к-та физ.-мат. н. Самара, 2013. С. 339. (Самарск. гос. аэрокосм. ун-т им. ак. С.П. Королева (нац. исслед.))

[10] Фильченков А. А, Тулупьев А. Л. Связность и ацикличность первичной структуры алгебраической байесовской сети // Вестник СПбГУ. Серия 1: Математика, Механика, Астрономия. № 1. С. 110–118. 2013.

[11] Фильченков А. А., Тулупьев А. Л., Сироткин А. В. Минимальные графы смежности алгебраической байесовской сети: формализация основ синтеза и автоматического обучения // Нечеткие системы и мягкие вычисления. 2011. Т. 6. № 2. С. 145–163.

[12] Фильченков А. А., Фроленков К. В., Сироткин А. В., Тулупьев А. Л. Система алгоритмов синтеза подмножеств минимальных графов смежности // Труды СПИИРАН. 2013. Вып. 4 № 27.

[13] Mal'chevskaya E. A., Berezin A. I., Zolotin A. A., Tulupyev A. L. Algebraic Bayesian Networks: Local Probabilistic-Logic Inference Machine Architecture and Set of Minimal Joint Graphs // Proceedings of the First International Scientific Conference «Intelligent Information Technologies for Industry» (IITI'16). Vol. 2. Sochi: Springer, 2016. PP. 69–79.