Компьютерные инструменты в образовании, 2015
№ 6: 3–18
УДК: 004.8
http://ipo.spb.ru/journal

информатика

ИНКРЕМЕНТАЛЬНЫЙ АЛГОРИТМ СИНТЕЗА МИНИМАЛЬНОГО ГРАФА СМЕЖНОСТИ*

Левенец Даниил Григорьевич, Зотов Михаил Анатольевич,
Тулупьев Александр Львович

Аннотация

В статье предложен инкрементальный алгоритм, ускоряющий синтез минимального графа смежности по исходному минимальному графу смежности и новой вершине с заданной допустимой нагрузкой; доказана корректность работы такого алгоритма. Выполнен сравнительный анализ статистических оценок сложности реализаций предложенного алгоритма и двух известных алгоритмов синтеза минимального графа смежности: жадного и прямого; результаты вычислительных экспериментов представлены в виде графиков, снабжены комментариями и выводом. В целом, на наборах со средним и большим числом вершин инкрементальный алгоритм в 3–6 раз оказался быстрее, чем прямой, и в 10–50 раз — быстрее, чем жадный. На наборах с небольшим числом вершин скорость алгоритмов отличается не сильно. Отмечено, что, чем больше в наборе вершин с нагрузкой 5–9 символов, тем более велик разброс отношения скоростей сравниваемых алгоритмов. В статье также достигнуты дидактические цели: в контексте синтеза минимальных графов смежности продемонстрировано обоснованное применение методов инкрементализации алгоритмов, статистического анализа скорости работы реализаций алгоритмов, визуализации данных, полученных в результате проведения вычислительных экспериментов.

Ключевые слова: *граф смежности, вероятностные графические модели, инкрементальный алгоритм, статистическая оценка сложности, структурное обучение, машинное обучение.*

1. ВВЕДЕНИЕ

Графы смежности и деревья смежности (то есть графы смежности без циклов) интенсивно используются в теории байесовских сетей [6, 9, 25], теории алгебраических байесовских сетей [4, 6–10] и некоторых других. Более подробные сведения о приложениях содержатся в [1, 2, 6], там же описаны и исследованы жадный и прямой алгоритмы синтеза не просто графа смежности, а минимального графа смежности. Хотя основные определения будут даны в разделе 2, отметим, что граф смежности — это ненаправленный граф, причем:

*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».

1) в его вершинах стоят несовпадающие и не поглощающие друг друга нагрузки (небольшие наборы символов без повторов над некоторым алфавитом);
2) он магистрально связен, то есть если у пары вершин этого графа нагрузки пересекаются, то они должны быть соединены магистралью [15–17] — путем, где нагрузка каждой промежуточной вершины содержит пересечение нагрузок концов пути, то есть той самой пары вершин. В частности, путь может содержать одно ребро [6].

Такой объект оказался необходим, в частности, потому, что ряд алгоритмов обработки алгебраических байесовских сетей требует, чтобы вторичной глобальной структурой такой сети был бы ациклический минимальный граф смежности [4, 10, 16]. Кроме того, минимальный граф смежности легче визуализировать, равно как и легче выявлять скрытые закономерности, им представленные. Наконец, минимальный граф смежности, не говоря уже о графе смежности, представляет собой достаточно сложную систему.

При сложившейся практике обмена данными между программным компонентом, реализующим графический пользовательский интерфейс, и «блоком вычислений» — программным компонентом, обеспечивающим построение сложной системы из заданных составляющих, типичный сценарий работы предполагает формирование набора составляющих, отправку набора в блок вычислений, формирование сложной системы, отправку данных в графический пользовательский интерфейс для визуального представления сформированной системы. Особенно это справедливо для клиент-серверных систем с так называемым тонким клиентом.

Однако этот типичный сценарий ведет ко все более длительному ожиданию оператора, когда набор составляющих растет, поскольку зачастую вычислительная сложность алгоритмов синтеза экспоненциально зависит от объема исходных данных или оказывается еще более большой.

Одним из возможных путей решения или, по крайней мере, смягчения обозначенной проблемы, является инкрементализация алгоритма синтеза, которая нацелена на использование того, что было построено на предшествующих шагах работы с программой.

Иначе говоря, на практике при работе с большим набором данных, в котором имеются связи, зависимости и отношения, часто ставится задача дополнения такого набора данных новым объектом таким образом, чтобы имеющиеся ранее связи, зависимости и отношения либо сохранялись прежними, либо время, затрачиваемое на их перестроение, было минимальным [20, 21, 23, 24, 26, 27]. Трудозатраты на выполнение таких операций играют важную, а порой и критическую роль, когда речь идет о real-time-системах или о системах, для которых важна высокая степень доступности. Таким образом, возникает вопрос — как избежать полного перестроения системы после каждого дополнения исходного набора данных новыми небольшими по сравнению со всем набором данными?

В рассматриваемом случае — примере минимального графа смежности как сложной системы — осуществляется попытка использовать уже построенный минимальный граф смежности и дополнить его новой вершиной с допустимой нагрузкой, то есть нагрузкой, которая не совпадает с нагрузкой никакой другой вершины, не поглощается нагрузками и не поглощает нагрузки никаких других вершин. Обратимся к более детальному рассмотрению существующего положения вещей и средств его улучшения.

Для того чтобы граф являлся графом смежности, необходимо поддерживать выполнение определённых условий (см. определение из раздела 2), результат проверки которых зависит от текущего набора данных; также, чтобы граф смежности являлся *минималь-*

ным графом смежности, необходимо поддерживать выполнение других условий, которые, в свою очередь, также зависят от текущего набора данных.

В отношении предложенных ранее жадного и прямого алгоритмов синтеза минимального графа смежности были построены теоретические оценки сложности [11], равно как проведен статистический анализ сложности этих двух конкурирующих алгоритмов [2]. Результаты статистического анализа отношений скорости работы двух алгоритмов позволили выделить три поддиапазона мощности наборов вершин графов смежности: в поддиапазоне 5–35 жадный алгоритм работает существенно быстрее прямого, в поддиапазоне 60–105 прямой алгоритм работает существенно быстрее жадного, а в поддиапазоне 35–60 выигрыш в скорости зависит от конкретного набора данных [2]. Кроме того, было установлено, что в диапазоне 5–60 имеется некоторое число статистических выбросов, сигнализирующих об особенностях в соответствующих наборах исходных данных [3]. Наконец, следует учесть, что на достаточно большом множестве наборов нагрузок, а именно когда мощность набора превышает некоторый уровень, оба алгоритма работают долго, в частности, и с субъективной точки зрения оператора тоже.

Именно поэтому особо актуальной задачей представляется добавление новой вершины v в уже существующий минимальный граф смежности $G = \langle V, E \rangle$ таким образом, чтобы граф $G' = \langle V \cup \{v\}, E' \rangle$ также оказался минимальным графом смежности, а время, затраченное на построение множества E', было минимальным, либо хотя бы приемлемым с точки зрения пользователя или решаемых задач машинного обучения.

Подводя итог, подчеркнем, что при добавлении новой вершины в минимальный граф смежности синтез множества ребер может занимать значительное время, в то время как множество ребер, полученное на выходе, будет слабо отличаться от исходного множества. Таким образом, актуальной *научно-технической целью* настоящей работы представляется ускорение синтеза минимального графа смежности за счет модернизации одного из существующих алгоритмов с помощью метода инкрементализации. Модернизированный алгоритм должен добавлять к уже существующему минимальному графу смежности вершину так, чтобы полученный граф оставался минимальным графом смежности, а время, затраченное на добавление новой вершины, было меньше времени, затраченного на генерацию нового минимального графа смежности «с нуля» по всей совокупности вершин $V' = V \cup \{v\}$. Указанная цель включает в себя компаративный анализ статистической сложности реализаций нового инкрементального алгоритма и двух алгоритмов, описанных ранее: жадного и прямого [2, 3].

Работа также имеет соотнесенные с основной целью *дидактические цели*. Во-первых, осветить на конкретном — и актуальном в сфере наукоемких ИТ — примере применение метода инкрементализации алгоритмов, обрабатывающих или синтезирующих сложные системы. В качестве примера сложной системы выбран минимальный граф смежности, применяющийся в теории байесовских сетей в качестве представления вторичной глобальной структуры алгебраических байесовских сетей [6, 9]. Во-вторых, показать мотивированное [2, 3] применение статистического подхода к сравнительной оценке скорости работы реализаций алгоритмов. Их применение обусловлено тем, что теоретические оценки сложности обсуждаемых алгоритмов не очевидны, зависят от скрытых особенностей исходных данных и не вычисляются по ним непосредственно. В-третьих, продемонстрировать приемы визуализации статистических данных заметного объема, снабженные последующей интерпретацией результатов.

2. ОПРЕДЕЛЕНИЯ И ОБОЗНАЧЕНИЯ

Воспользуемся системой терминов и обозначений, сложившихся в [4, 6–9].

Пусть задан конечный алфавит символов A, а непустые множества символов (без повторов) — слова — рассматриваются как возможные значения нагрузок вершин графов (в основном, графов смежности) и их ребер. Пусть имеется набор вершин $V = \{v_1, v_2, \ldots, v_n\}$ и нагрузки $W = \{W_1, W_2, \ldots, W_n\}$, причем $\forall u \in V \;\; W_u$ является нагрузкой для вершины u. Контекст статьи позволяет для удобства отождествлять вершины и веса как соотносящиеся взаимно однозначно (по-моему, поэтому мы станем использовать обозначения u и W_u взаимозаменяемо.

Определение 1. Назовем неориентированный граф $G = \langle V, E \rangle$ графом смежности, если он удовлетворяет следующим условиям:

1. $\forall u, v \in V$, таких что $W_u \cap W_v \neq \emptyset$, существует некоторый путь P в графе G такой, что для каждой вершины $s \in P$ справедливо утверждение
$$W_u \cap W_v \subseteq W_s;$$
2. $\forall \{u, v\} \in E \;\; W_u \cap W_v \neq \emptyset$.

Графы смежности с минимальным и максимальным числом ребер мы будем называть *минимальным графом смежности* и *максимальным графом смежности* соответственно.

Определение 2. Пересечение нагрузок двух вершин $W_u \cap W_v$ будем называть *сепаратором*.

Определение 3. Число элементов в заданном сепараторе назовем *мощностью сепаратора*.

Определение 4. Будем называть *списком* набор объектов, доступных по индексу.

Введем ряд дополнительных обозначений, которые понадобятся нам при описании алгоритмов.

PathExists(G, edge, nodeBegin, nodeEnd) — функция, которая определяет, связаны ли магистрально вершины nodeBegin и nodeEnd в графе $G' = \langle V, E' \rangle$, причем $G = \langle V, E \rangle$ и $E' = \{\{p, q\} : \{p, q\} \in E \;\&\; \{p, q\} \neq \text{edge}\}$. [2, 4, 5]

edge.First, edge.Second — вершины ребра edge.

edge.RemoveAllowed — метка, которая показывает, возможно ли удаление ребра edge или нет. Изначальное значение — true.

3. ИНКРЕМЕНТАЛЬНЫЙ АЛГОРИТМ ДОБАВЛЕНИЯ ВЕРШИНЫ В МИНИМАЛЬНЫЙ ГРАФ СМЕЖНОСТИ

Задан минимальный граф смежности G, на вход алгоритму поступает указанный граф и новая вершина. Требуется построить новый минимальный граф смежности, такой что множество вершин состоит из объединения множества вершин V графа G и новой вершины. При этом для простоты изложения предполагаем, что набор нагрузок остается корректным: нагрузка новой вершины допустима, то есть её нагрузка и нагрузка любой вершины из множества V не совпадают, а также нагрузка новой вершины не поглощает полностью нагрузку любой вершины из множества V и не поглощается такими нагрузками. Используя формальные обозначения, то же самое запишем следующим образом.

$G = \langle V, E \rangle$ — граф, подаваемый на вход алгоритму. v — новая вершина, подаваемая на вход алгоритму. $G' = \langle V \cup \{v\}, E' \rangle$, где E' — множество ребер минимального графа смежности G'. Как отмечалось ранее, инкрементальный алгоритм берет за основу построения минимального графа смежности G' множество ребер E, благодаря такому подходу предполагается, что инкрементальный алгоритм будет работать быстрее жадного (прямого), так как большая или даже бóльшая часть связей (ребер) останется без изменений.

Напомним, что в контексте данной работы обозначения вершины и ее нагрузки совпадают. Такой подход будет удобен для чтения и понимания графиков, представленных ниже, а также в разделе о корректности алгоритма.

3.1. Описание алгоритма

На листинге 1 приведен псевдокод инкрементального алгоритма добавления новой вершины в минимальный граф смежности

Algorithm 1 Инкрементальный алгоритм построения минимального графа смежности.

input: $G = \langle V, E \rangle$, v
output: $G' = \langle V', E' \rangle$

```
 1: function SimpleIncremental
 2:     E' = E
 3:     V' = V ∪ {v}
 4:
 5:     foreach (u in V)
 6:         if ((W_u ∩ W_v) ≠ ∅) then
 7:             E' = E' ∪ {u, v}
 8:
 9:     G' =< V', E' >
10:
11:     while (true)
12:         edge = ∅
13:
14:         foreach(e in E')
15:             if (e.RemoveAllowed) then
16:                 edge = e
17:                 break foreach //Выход из foreach
18:
19:         if (edge == ∅) then
20:             break while //Выход из while
21:
22:         if (PathExists(G', edge, edge.First, edge.Second)) then
23:             E' = E' \ {edge}
24:         else
25:             edge.RemoveAllowed = false
26:
27:     return G'
```

Приведенный алгоритм добавляет новую вершину в существующий минимальный граф смежности G, а на выходе отдаёт минимальный граф смежности G' с добавленной новой вершиной.

Алгоритм состоит из двух частей. В первой части (строки 2–7) во вторичную структуру АБС добавляются все допустимые ребра. Во второй (строки 8–19) — происходит удале-

ние тех ребер, отсутствие которых не приведет к нарушению магистральной связности вершин графа, другими словами — синтезируется минимальный граф смежности. Рассмотрим алгоритм более подробно.

В строке 3 в множество вершин V', состоящее в точности из вершин V исходного графа G, добавляется новая вершина v. В строке 4 для каждой вершины из множества V и новой вершины v проверяется наличие непустого сепаратора. Если такой сепаратор есть, то в множество E' добавляется соответствующее ребро.

После добавления новой вершины и новых ребер формируется граф G', далее из него необходимо удалить те ребра, отсутствие которых не приведет к нарушению магистральных связей (строки 8 –19). В результате такого преобразования граф G' станет минимальным графом смежности. Отметим, что ребра удаляются с помощью жадного алгоритма, описанного в статьях [3–5].

Приведенный алгоритм позволяет добавлять новую вершину v в уже существующий минимальный граф смежности G. Отметим, что в указанном алгоритме для добавления новой вершины не требуется заново синтезировать множество ребер искомого графа смежности, так как инкрементальный алгоритм учитывает и дополняет полученные ранее результаты.

Посмотрим более пристально и с более общей точки зрения на то, как и в чем проявились преимущества инкрементального алгоритма.

Отметим, что в инкрементальном алгоритме были использованы результаты и объекты, построенные на предыдущих шагах. В данном случае это — минимальный граф смежности, который предстоит дополнить новой вершиной, то есть собственно основной результат предшествующего запуска алгоритма синтеза. В других случаях могут оказаться доступны и вспомогательные конструкции или объекты, которые использовались при построении основного результата. Например, это можно ожидать в случае, когда нужно построить не один минимальный граф смежности, а все возможные минимальные графы смежности [11–15, 17].

Кроме того, учтем, что потребовалось строить не весь набор ребер максимального графа смежности, а лишь те, которые исходят из новой вершины. При этом экономия в числе операций может возникнуть и, исключая особые, специально подобранные случаи, как правило, возникает за счет двух моментов: во-первых, требуется построить меньшее число ребер, во-вторых, последующий перебор для исключения избыточных ребер идет по меньшему их числу.

Наконец, экономия может проявиться и за счет того, что из пользовательского интерфейса пересылаются сведения об одной новой вершине, а не о всей их совокупности. Конечно, в случае нагрузок графа смежности в той форме, в которой они описаны выше, эффект экономии не будет так заметен. Но если вершина, в свою очередь, определяется сложным объемным набором данных, оператор почувствует ускорение в обработке его команды.

3.2. Корректность алгоритма

3.2.1. Алгоритм синтезирует граф смежности

При добавлении сразу всех допустимых ребер, исходящих из новой вершины v, в граф смежности получается новый граф смежности, следовательно, после выполнения первой части алгоритма (строки 2 –6) граф G' также будет графом смежности, при этом

допустимость этих ребер обуславливается проверкой, осуществляемой в строке 5. Проанализируем корректность этой части алгоритма более подробно.

Теорема. Пусть $G = \langle V, E \rangle$ — минимальный граф смежности, v — добавляемая (новая) вершина, $V' = V \cup \{v\}$, а $E' = E \cup \{\{v, w\} : w \in V \ \& \ W_v \cap W_w \neq \emptyset\}$, тогда $G' = \langle V', E' \rangle$ — граф смежности.

Доказательство: Возьмем любые две вершины s и $r \in G'$. Рассмотрим два случая.

Сепаратор s и $r \neq \emptyset$, тогда докажем, что в G' существует путь между этими вершинами. Предположим, что обе вершины принадлежат исходному графу(s и $r \in G$), тогда, по определению графа смежности, между ними существует некоторый путь. Если же одна из вершин не принадлежит исходному графу, то она совпадает с вершиной v ($s = v$ или $r = v$), следовательно, по определению множества $E' = E \cup \{\{v, w\} : w \in V \ \& \ W_v \cap W_w \neq \emptyset\}$, между ними существует ребро, которое и является путем с нужными свойствами, то есть магистралью.

Сепаратор s и r равен $\emptyset$, тогда докажем, что в G' не существует магистрали между этими вершинами. Предположим, что обе вершины принадлежат исходному графу(s и $r \in G$), тогда, по определению графа смежности, между ними не существует ребра. Если же одна из вершин не принадлежит исходному графу, то она совпадает с вершиной v ($s = v$ или $r = v$), следовательно, по определению множества $E' = E \cup \{\{v, w\} : w \in V \ \& \ W_v \cap W_w \neq \emptyset\}$, — между ними не существует ребра.

Таким образом, любая пара вершин, пересечение нагрузок которых не пусто, соединены магистралью, а вершины, нагрузки которых не пересекаются, не соединены ребром. Граф, обладающий таким свойством, является графом смежности. Также следует отметить, что полученный на первом шаге граф не обязательно является минимальным графом смежности. Приведем пример.

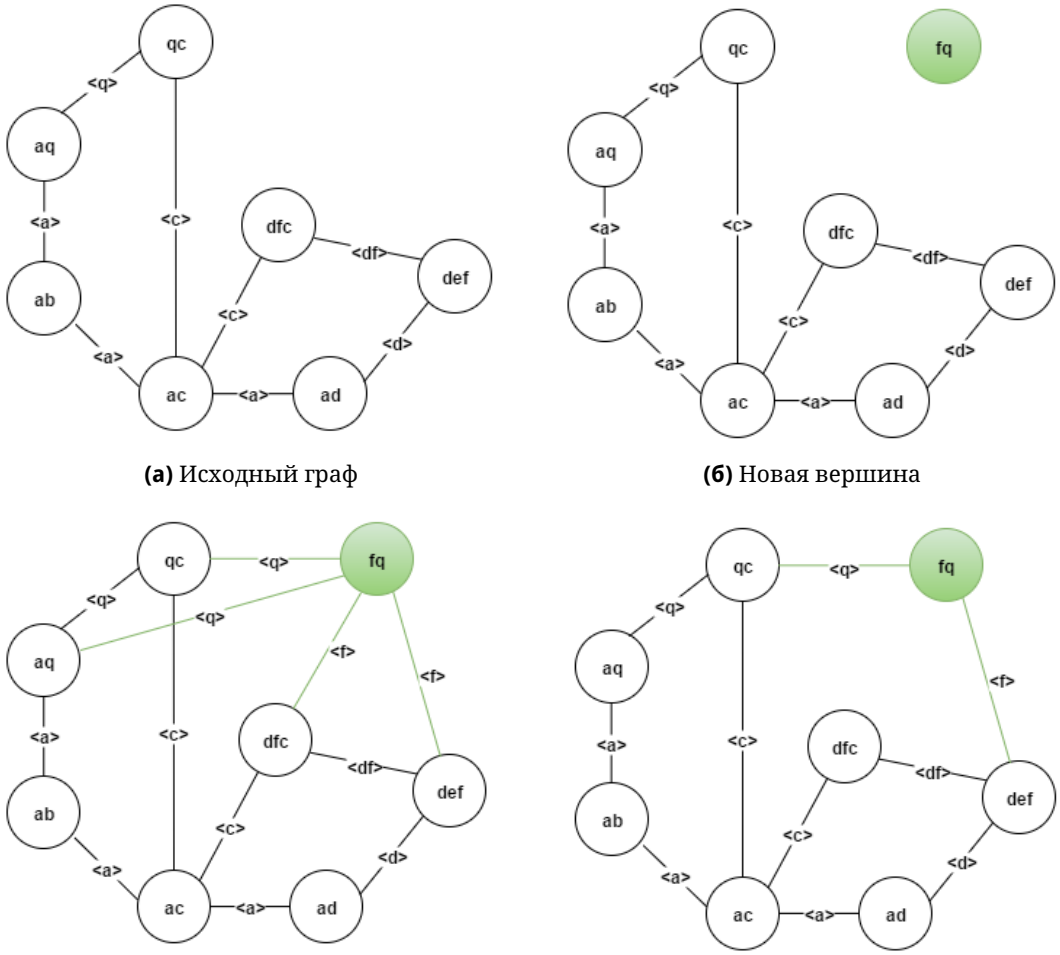
В минимальный граф смежности G (рис. 1а) добавляется новая вершина v с нагрузкой $\{f, q\}$ (рис. 1б). На первом шаге инкрементального алгоритма в граф добавляются несколько ребер с нагрузками $\{q\}$ и $\{f\}$ (рис. 1в). Заметим, что полученный граф G' не является минимальным графом смежности, так как удаление ребра $\{aq, fq\}$ или любого другого ребра с нагрузкой $\{q\}$ не приводит к нарушению свойств графа смежности (рис. 1г).

Следует напомнить о предположении, что набор нагрузок должен быть корректен — это означает, что нагрузка v не равна нагрузке никакой другой вершины, в нее не входят полностью нагрузки других вершин, и она сама не поглощается нагрузками дугих вершин. Случаи с некорректными нагрузками нужно рассматривать отдельно, что не входит в цели настоящей работы.

3.2.2. Алгоритм синтезирует минимальный граф смежности

Удаление ребра при сохранении магистральности есть не что иное, как переход от одного графа смежности к другому с уменьшением числа ребер. В статьях [4, 5] было доказано, что множество графов смежности формирует матроид.

По свойствам матроида, последовательность допустимых удалений ребер в графе смежности сходится, а полученный в результате завершившейся последовательности допустимых удалений граф будет минимальным графом смежности. Действительно, как было показано ранее, после выполнения первой части алгоритма граф G' является графом смежности, а во второй части алгоритма происходят удаления допустимых ребер. Таким образом, по свойствам матроида, алгоритм из листинга 1 на выходе форми-



(а) Исходный граф **(б)** Новая вершина

(в) Новые ребра **(г)** Новый минимальный граф смежности

Рис. 1. Интеграция новой вершины в граф смежности с сохранением его минимальности

рует минимальный граф смежности, включающий новую вершину v, что и требовалось доказать.

4. ОПИСАНИЕ ВЫЧИСЛИТЕЛЬНЫХ ЭКСПЕРИМЕНТОВ

Подробное описание методики сравнения статистической сложности реализаций алгоритмов было приведено в статье (ссылка), и в настоящей работе мы будем ей следовать, сравнивая попарно реализации жадного и инкрементального, а также прямого и инкрементального алгоритмов.

Перечисленные выше алгоритмы были реализованы на языке программирования C#. Вкратце, основная идея эксперимента для вычисления отдельного показателя относительной скорости воплощена в следующих шагах.

1. Генерировался набор вершин V.
2. По набору вершин V синтезировался минимальный граф смежности G, который на шаге 3 будет выступать в качестве системы, в которую мы хотим добавить новые данные.

3. В граф G с помощью инкрементального алгоритма добавлялась вершина v. Время выполнения алгоритма фиксировалось.
4. По множеству вершин $V \cup \{v\}$, где v — новая вершина, строился минимальный граф смежности с помощью жадного (прямого) алгоритма. Время выполнения жадного (прямого) алгоритма также фиксировалось.
5. По результатам вычислялось отношение величин времени работы прямого (жадного) и инкрементального алгоритмов.

Однако этот эксперимент для получения обоснованной статистической оценки отношений времени работы модифицировался следующим образом.

Система, в которую добавлялись новые данные жадным (прямым) алгоритмом, заново перестраивалась, а в случае инкрементального алгоритма — достраивалась.

Следует оговорить, что влияние на время работы алгоритма могут оказывать некоторые случайные факторы, например текущее состояние ОС или загруженность центрального процессора, на котором запущено приложение, вычисляющее данную оценку. Также время работы алгоритма зависит от набора данных. Чтобы минимизировать влияние случайных факторов и описать распределение отношений скорости работы алгоритмов на наборах данных, воспользуемся методикой, описанной в статьях [2, 3].

Сначала генерируются различные наборы входных данных (в наших экспериментах — 50), каждый из которых передается алгоритмам синтеза минимального графа смежности несколько раз (в наших экспериментах — 60). Значения скоростей алгоритмов, полученных на *одном и том же* наборе данных, усредняются. После этого рассматриваются отношения усредненных скоростей двух алгоримов, полученных на каждом из *различных, но одинаковой мощности* наборах данных. По сформированной *выборке* отношений скоростей вычисляются значения статистик для одного конкретного числа вершин в графе. Описанные действия повторяются для всех мощностей из набора 5, 10, …, 110 (шаг равен 5); в итоге получаются значения статистик для каждой из мощностей. На графиках представлены следующие статистики: R_g — медиана, Q_1, Q_3 — соответственно первый и третий квартили, D_1, D_9 — первый и девятый децили, R^+, R^- — верхняя и нижняя границы 97%-го доверительного интервала.

5. РЕЗУЛЬТАТЫ ЭКСПЕРИМЕНТОВ

На рисунках 2–8 представлены графики указанных статистик относительного времени выполнения жадного и инкрементального алгоритмов на разных наборах данных. На горизонтальной оси указано число вершин в графе, а на вертикальной — величина статистики. Статистики характеризуют выборку значений случайной величины — отношения времени работы жадного (прямого) алгоритма к времени работы инкрементального:

$$\text{статистика характеризует } \frac{\text{время работы жадного (прямого) алгоритма}}{\text{временя работы инкрементального}}.$$

Поясним, что время работы прямого (жадного) алгоритма делится на время работы инкрементального алгоритма, или, что одно и то же, скорость работы прямого (жадного) алгоритма делится на скорость работы инкрементального алгоритма.

Результаты экспериментов позволяют сделать вывод, что в поддиапазонах мощностей наборов вершин 10–110 прямой и жадный алгоритмы значительно уступают по скорости инкрементальному. Жадный алгоритм работает дольше инкрементального на

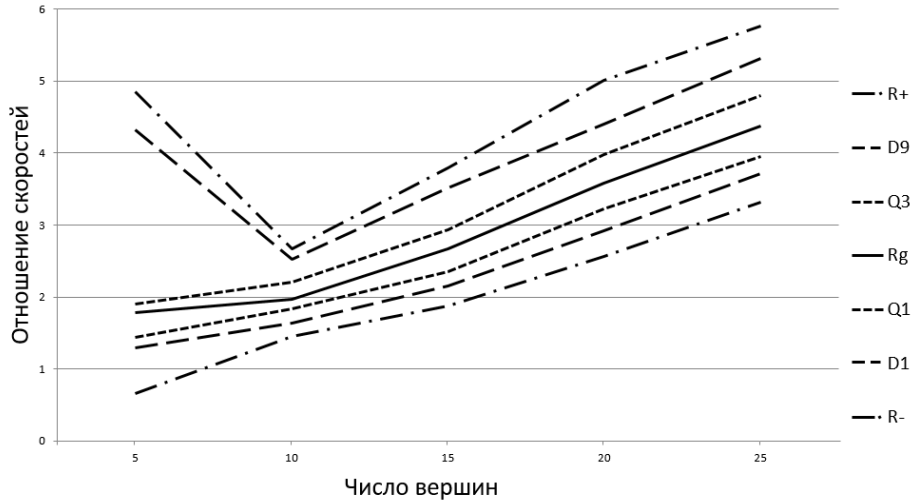


Рис. 2. Жадный и инкрементальный алгоритмы. Алфавит 52 символа, вершины 2–4-го порядков, диапазон 5–25

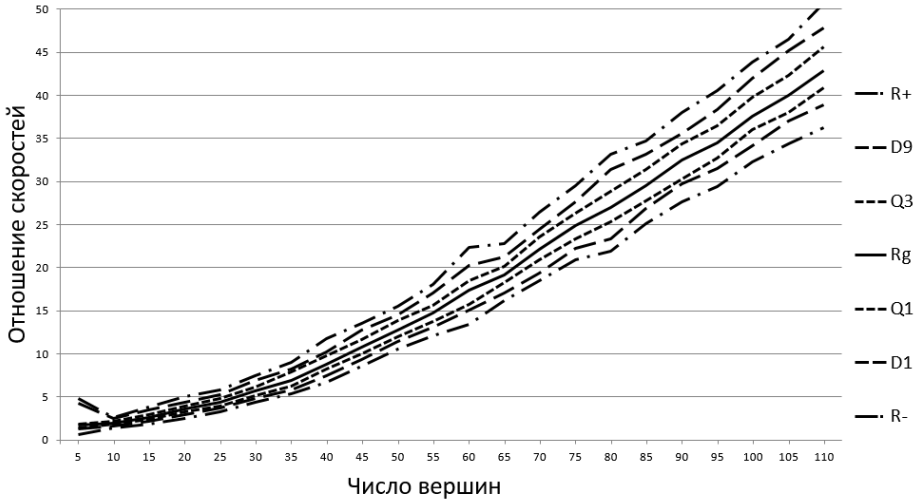


Рис. 3. Жадный и инкрементальный алгоритмы. Алфавит 52 символа, вершины 2–4-го порядков, диапазон 10–110

промежутке 10–50 в разы, а на промежутке от 50 и выше — в десятки раз. Инкрементальный алгоритм работает быстрее, чем прямой на всем промежутке измерений, а с 10 и выше — в 3–5 раз. С ростом числа нагрузок в наборе ускорение инкрементального алгоритма по отношению к прямому растет постепенно, но заметно: это подтверждают все линии-графики статистик.

В отношении скоростей инкрементального и жадного алгоритмов поддипазон 5–10 содержит особенность. На рис. 2–3 видно, что нижняя граница доверительного интервала находится ниже отметки 1, поэтому нельзя делать вывод о том, что инкрементальный алгоритм превосходит в скорости работы жадный алгоритм на всём наборе данных. Отметим также, что время синтеза минимального графа смежности для 5–10 вершин незначительно для любого из трех алгоритмов.

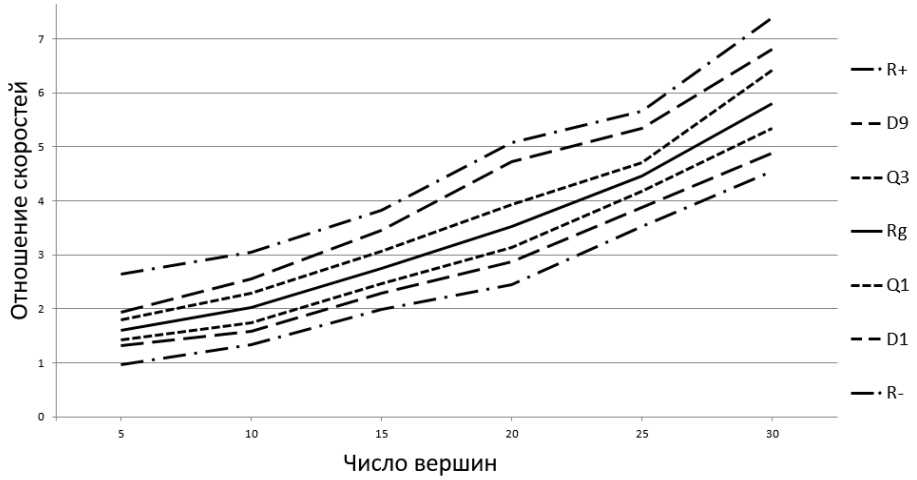


Рис. 4. Жадный и инкрементальный алгоритмы. Алфавит 52 символа,вершины 2–4-го порядков, диапазон 5–30. 20% вершин 5–9 порядков

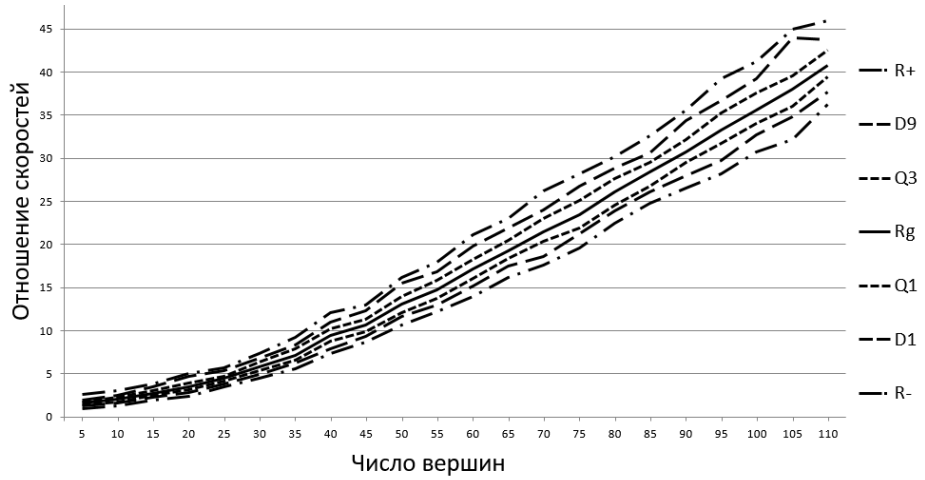


Рис. 5. Жадный и инкрементальный алгоритмы. Алфавит 52 символа, вершины 2–4-го порядков, диапазон 10–110. 20% вершин 5–9 порядков

Несмотря на то, что выявленное исключение представляет теоретический интерес и может послужить отправной точкой для улучшения результата или выявления ранее скрытых закономерностей, с практической точки зрения, в данный момент выбор алгоритма синтеза для диапазона 5–10 не принципиален из-за незначительности временных затрат. Следует также акцентировать внимание на двух тонкостях: жадный алгоритм наращивает свое отставание при увеличении числа вершин, а прямой алгоритм — при увеличении числа вершин большого порядка, то есть величина статистики зависит от скрытых свойств набора данных. Наконец, необходимо подчеркнуть «неплавное» поведение графиков статистик, что говорит о наличии в них значительных колебаний при росте доли вершин большого порядка.

Это наблюдение позволяет сформулировать две гипотезы для дальнейших исследований:

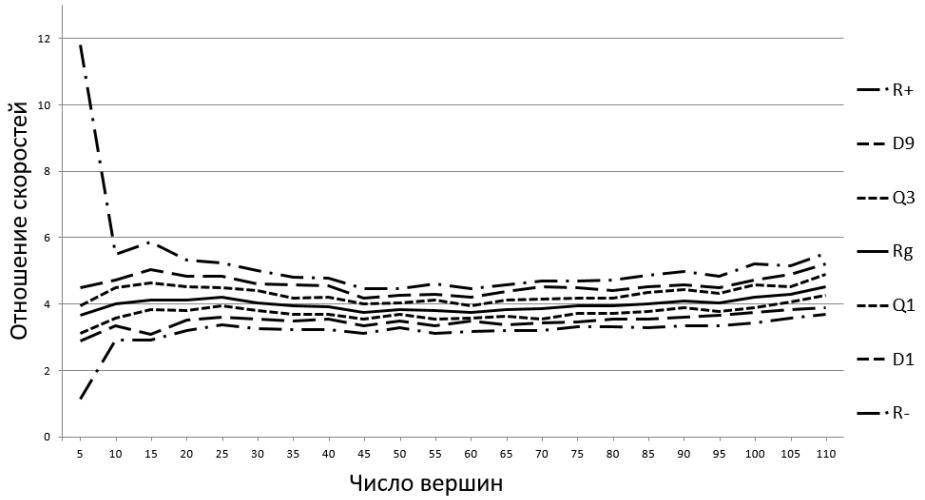


Рис. 6. Прямой и инкрементальный алгоритмы. Алфавит 52 символа, вершины 2–4-го порядков

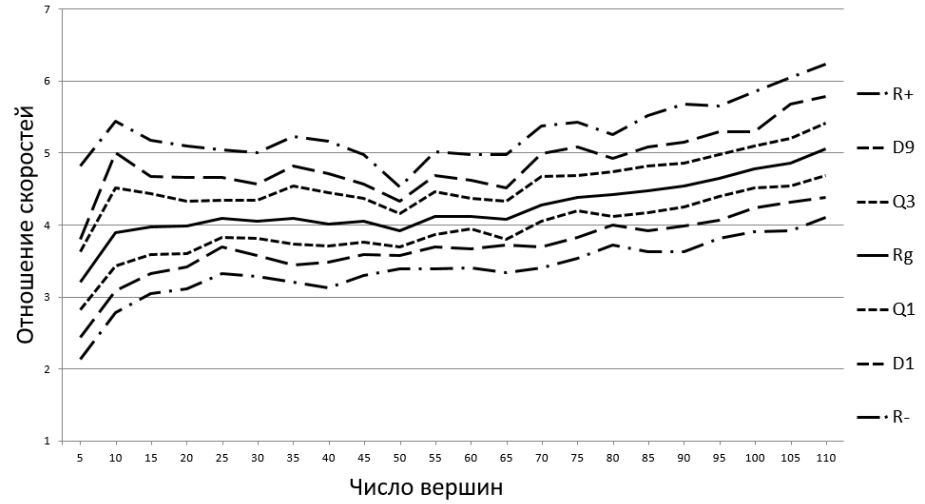


Рис. 7. Прямой и инкрементальный алгоритмы. Алфавит 52 символа, вершины 2–4-го порядков. 10% вершин 5–9

порядков

1) выборка для каждого сочетания параметров должна быть более многочисленной для увеличения вероятности попадания наборов «с особенностями» в текущий набор;
2) большая доля вершин с более «тяжелой» нагрузкой ведет к появлению структурных и иных особенностей графов смежности или минимальных графов смежности, которые существенным образом сказываются на величине статистической оценки сложности программных реализаций трех алгоритмов, рассматриваемых в настоящей статье.

Если указанные гипотезы верны, то такие особенности потребуется выявить, изучить и использовать в дальнейшем для ускорения синтеза минимальных графов смежности.

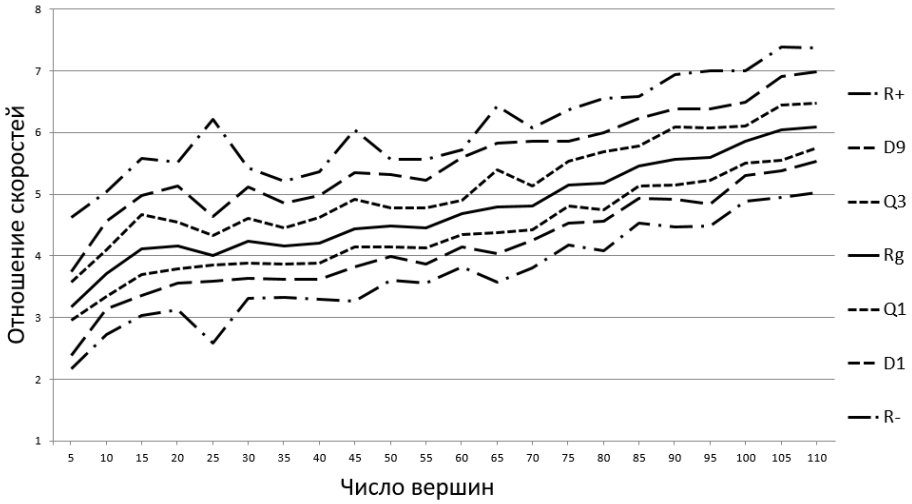


Рис. 8. Прямой и инкрементальный алгоритмы. Алфавит 52 символа, вершины 2–4-го порядков. 20% вершин 5–9 порядков

6. ЗАКЛЮЧЕНИЕ

В статье предложен инкрементальный алгоритм, ускоряющий синтез минимального графа смежности по исходному минимальному графу смежности и новой вершине с заданной допустимой нагрузкой; доказана корректность работы такого алгоритма. Основной выигрыш в скорости работы достигается за счет использования уже построенного на предшествующем этапе минимального графа смежности, а также за счет сокращения числа перебираемых ребер для исключения избыточных, чтобы из полученного включением новой вершины и новых ребер графа смежности получить минимальный граф смежности. Отметим, что рассматривались лишь допустимые нагрузки, то есть такие, которые не совпадают с нагрузками других вершин, не поглощают такие нагрузки и не поглощаются ими. Доказана корректность работы инкрементального алгоритма.

Инкрементальный алгоритм особенно удобен для интерактивных программных систем, когда минимальный граф смежности строится в диалоге с пользователем. Последний ожидает от системы достаточного быстродействия, полагая, что небольшое изменение в данных (добавляется одна вершина), влечет умеренное изменение минимального графа смежности.

Выполнен сравнительный анализ статистических оценок сложности реализаций предложенного инкрементального алгоритма и двух ранее разработанных алгоритмов синтеза минимального графа смежности: жадного и прямого [2, 3].

На выбранном для вычислительных экспериментов диапазоне 5–110 числа нагрузок, над которыми строится граф, инкрементальный алгоритм стабильно, как минимум в 2–5 раз превосходит прямой, а если опираться на среднее отношение времени работы, то в 3–6 раз.

Над диапазоном 10–50 инкрементальный алгоритм работает быстрее, чем жадный алгоритм, в разы, а над диапазоном 50–110 — в десятки раз. В начале же диапазона 5–10 невозможно достоверно утверждать, что *в целом* какой-то алгоритм превосходит другой, поскольку линия, отвечающая за долю 97% («два сигма»), идет ниже, чем линия, соответствующая отношению скоростей, равному единице. Однако на некоторых наборах дан-

ных (более, чем на 90% таких наборов) инкрементальный алгоритм все -таки работает быстрее, чем жадный алгоритм.

Разброс отношений скоростей сравниваемых алгоритмов тем больше, чем больше доля более «тяжелых» нагрузок в исходном наборе данных.

Выявленные особенности составляют основу для формирования целей дальнейших исследований, которые могут привести к последующему совершенствованию алгоритмов и их реализаций, что, в свою очередь, позволит еще больше ускорить синтез минимальных графов смежности.

В статье также достигнуты дидактические цели: в контексте синтеза минимальных графов смежности продемонстрировано обоснованное применение методов инкрементализации алгоритмов, статистического компаративного анализа скорости работы программных реализаций, визуализации данных, полученных в результате проведения вычислительных экспериментов.

Список литературы

1. *Золотин А.А., Тулупьев А.Л., Сироткин А.В.* Матрично-векторные алгоритмы нормировки для локального апостериорного вывода в алгебраических байесовских сетях // Научно-технический вестник информационных технологий, механики и оптики, 2015. Т. 15. № 1. С. 78–85.
2. *Зотов М.А., Тулупьев А.Л.* Синтез вторичной структуры алгебраических байесовских сетей // Компьютерные инструменты в образовании, 2015. № 1. С. 3–16.
3. *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.
4. *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.
5. *Опарин В.В., Фильченков А.А., Сироткин А.В., Тулупьев А.Л.* Матроидное представление семейства графов смежности над набором фрагментов знаний // Научно-технический вестник Санкт-Петербургского государственного университета информационных технологий, механики и оптики, 2010. № 4(68). С. 73–76.
6. *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).
7. *Тулупьев А.Л.* Дерево смежности с идеалами конъюнктов как ациклическая алгебраическая байесовская сеть // Тр. СПИИРАН. Вып. 3, т. 1. СПб.: Наука, 2006. С. 198–227.
8. *Тулупьев А.Л., Николенко С.И., Сироткин А.В.* Байесовские сети: логико-вероятностный подход. СПб.: Наука, 2006. С. 607.
9. *Тулупьев А.Л., Сироткин А.В., Николенко С.И.* Байесовские сети доверия: логико-вероятностный вывод в ациклических направленных графах. СПб.: Изд-во С.-Петерб. ун-та, 2009. С. 400.
10. *Тулупьев А.Л., Столяров Д.М., Ментюков М.В.* Представление локальной и глобальной структуры алгебраической байесовской сети в Java-приложениях // Труды СПИИРАН. 2007. Вып. 5. СПб.: Наука, 2007. С. 71–99.
11. *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.))
12. *Фильченков А.А., Мусина В.Ф., Тулупьев А.Л.* Алгоритм рандомизированного синтеза минимального графа смежности // Тр. СПИИРАН. 2013. Вып. 2(35). С. 221–234. ун-т.
13. *Фильченков А.А., Тулупьев А.Л.* Структурный анализ систем минимальных графов смежности // Тр. СПИИРАН. 2009. № 11. С. 104–129.

14. *Фильченков А.А., Тулупьев А.Л.* Анализ циклов в минимальных графах смежности алгебраических байесовских сетей // Тр. СПИИРАН. 2011. № 17. С. 151–173.
15. *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.
16. *Фильченков А.А., Тулупьев А.Л., Сироткин А.В.* Особенности анализа вторичной структуры алгебраической байесовской сети // Труды СПИИРАН. 2010. № 1 (12). С. 97–118.
17. *Фильченков А.А., Фроленков К.В., Сироткин А.В., Тулупьев А.Л.* Система алгоритмов синтеза подмножеств минимальных графов смежности // Труды СПИИРАН. 2013. № 4(27). С. 200–244.
18. *Шинкаренко В.И.* Зависимость временной эффективности алгоритмов и программ обработки больших объемов данных от их кэширования // Математические машины и системы. 2007. №2. С. 43–55.
19. *Barrett C., Marathe A., Marathe M., Cook D., Hicks G., Faber V., Srinivasan A., Sussmann Y., Thornquist H.* Statistical Analysis of Algorithms: A Case Study of Market-Clearing Mechanisms in the Power Industry. Journal of Graph Algorithms and Applications (JGAA). 2003. Vol. 7. P. 3–31.
20. *Gonzalez-Diaz R., Ion A., Jimenez M.J., Poyatos.R.* Incremental-Decremental Algorithm for Computing AT-Models and PersistentHomology // Real, P and DiazPernil, D and MolinaAbril, H and Berciano, A and Kropatsch, W, editor, Computer analysis of images and patterns:14th international conference, CAIP 2011, PT I, volume 6854 of LNCS, p.286–293, heidelberger platz3, D-14197 Berlin, Germany, 2011. springer-verlag berlin. 14thInternational Conference on Computer Analysis of Images and Patterns (CAIP),Seville, SPAIN, AUG 29-31, 2011.
21. Incremental algoritms [Электронный ресурс]. Режим доступа: URL: http://c2.com/cgi/wiki?IncrementalAlgorithms (дата обращения 01.11.2015).
22. *Jaynes E.T.* Bayesian Methods: General Background // Maximum-Entropy and Bayesian Methods in Applied Statistics, by J. H. Justice (ed.). Cambridge: Cambridge Univ. Press, 1986. P. 1–19.
23. *Jerzy S.* Respondek. Dynamic Data Structures in the Incremental Algorithms Operating on a Certain Class of Special Matrices. // Murgante, B and Misra, Sand Rocha, AMAC and Torre, C and Rocha, JG and Falcao, MI and Taniar,D and Apduhan, BO and Gervasi, O, editor, Computational scienceand its applications, part VI - ICCSA 2014, volume 8584 of Lecture Notes in Computer Science, p.171–185, Heidelberger platz 3, D-14197 Berlin, Germany, 2014. springer-verlag Berlin. 14-th International Conference on Computational Science and Its Applications (ICCSA), Guimaraes, Portugal, jun 30-jul 03, 2014.
24. *Kas M., Wachs M., Carley K.M., Carley.L.R.* Incremental Algorithm for Updating Betweenness Centrality in Dynamically Growing Networks. // Ozyer, T and Carrington, P, editor,2013 IEEE/ACM international conference on advances in social networks analysis and mining (ASONAM), p.39–46, 345 E 47TH ST, Newyork, NY 10017 USA, 2013. IEEE. IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), Niagara Falls, Canada, aug. 25-28, 2013.
25. *Pearl J.* Causality: Models, Reasoning, and Inference. Cambridge: Cambridge University Press, 2000. P. 400.
26. *Sariyuce A.E., Kaya K., Saule E., Umit V.* Catalyurek. Incremental Algorithms for Closeness Centrality. // Hu, X and Lin, TY and Raghavan, V and Wah, B and Yates, R and Fox, G and Shahabi, C and Smith, M and Yang, Q and Ghani, R and Fan, W and Lempel, R and Nambiar,R, editor,2013 IEEE International conference on big data, 345E 47-th st, Newyork, NY 10017 USA, 2013. IEEE. IEEE International Conference on Big Data (Big Data), Santa Clara, CA, oct. 06-09, 2013.
27. *Song X., Yu L., Sun H.* An Incremental Query Algorithm for Optimal Path Queries Under Traffic Jams. // Yu, F and Chen, W and Chen, Z and Yuan, J, editor, ISCSCT 2008: International Symposiumon Computer Science and Computational Technology, Proceedings, p.472–475, 10662 Los Vaqueros Circle, PO BOX3014, Los Alamitos, CA 90720-1264 USA, 2008. IEEE Computer SOC. International Symposium on Computer Science and Computational Technology, Shanghai, Peoples R China, dec.20-22, 2008.

AN INCREMENTAL ALGORITHM FOR MINIMAL JOINT GRAPH SYNTHESIS

Levenets D. G., Zotov M. A., Tulupyev A. L.

Abstract

The paper provides the description of an incremental algorithm that accelerates a minimal joint graph upgrade when a new node with a correct load is inserted. The algorithms correctness is proven. We compare relative performance statistical estimates for the new incremental algorithm and two known greedy and streightforward algorithms that implement minimal joint graph synthesis. The computational experiments results are represented with plots, discussed and summarized. For the middle and large sized sets of loads, the incremental algorithm is 3–6 times faster than the streightforward one and 10–50 times faster than the greedy one. For the small sized sets of loads, the algorithms permormance differs less dramatically. The relative performance variation of compared algorithms are the larger the higher fraction of loads with 5–9 symbols the set of loads has. The paper also reaches didactical goals: in the context of minimal joint grah synthesis, we demonstrate a rational application of the algorithm incrementation, an example of relative algorithmical performance statistical estimates, an approach to the visualisation of performance comparison compuational experiments.

Keywords: *joint graph, probabilistic graphical model, incremental algorithm, performance statistical estimate, structure learning, machine learning.*

Левенец Даниил Григорьевич,
студент 4 курса кафедры информатики математико-механического факультета СПбГУ,
daniel-levenets@yandex.ru

Зотов Михаил Анатольевич,
студент 5 курса кафедры системного программирования математико-механического факультета СПбГУ,
zotov1994@mail.ru

Тулупьев Александр Львович,
доктор физико-математических наук, доцент, заведующий лаб. ТиМПИ СПИИРАН; профессор кафедры информатики СПбГУ,
alexander.tulupyev@gmail.com