УДК 004.8+519.5

МИНИМИЗАЦИЯ ВРЕМЕНИ ВОССТАНОВЛЕНИЯ ГРАФА

Татаринов Е. А.
Институт прикладной математики и механики НАНУ
Лаборатория дискретной математики и прикладной алгебры
E-mail: Mdgerelo@yandex.ru

Аннотация

Татаринов Е. А. Минимизация времени восстановления графа. *Рассматривается алгоритм восстановления графа, основанный на методе построения неявной нумерации на его вершинах. Предлагается алгоритм, в котором минимизируется длина красного пути при восстановлении графа (обратных ребер).*

Введение

Анализ глобальных свойств графа при помощи блуждающих по нему агентов являются частной проблемой исследования поведения автоматов в лабиринтах [2 - 4]. Общая проблема анализа графа включает в себя ряд частных проблем, важнейшими из которых являются: проблема самолокализации, т.е. определении вершины графа, в которой первоначально находится автомат; проблема контроля графа (карты), т.е. проверки изоморфизма исследуемого графа и заданного графа - эталону (карты), и проблема полного восстановления графа. При этом автомат перемещается по дугам графа от вершины к вершине и воспринимает некоторую локальную информацию о нем и может отмечать элементы графа специальными метками (красками и / или камнями).

В данной работе оптимизируется алгоритм, предложенный в [6]. В [6] рассматривается задача восстановления конечного, неориентированного графа, без петель и кратных ребер при помощи агента, который перемещается по нему и изменяет метки на вершинах и ребрах графа [1]. Было показано, что верхняя оценка временной сложности алгоритма из [6] выражается через высоту дерева обхода в глубину и, в общем случае, будет квадратичной функцией от числа вершин в графе. В данной работе предлагается модификация алгоритма из [6], где верхняя оценка временной сложности выражается через длину наибольшего простого цикла в исследуемом графе, как это было сделано в [5]. Эта модификация так же основана на методе построения неявной нумерации на вершинах графа [10]. Оптимизация достигается за счет использования дополнительной краски и заключается в том, что верхняя оценка временной сложности , предложенного алгоритма, выражается через длину наибольшего цикла и на порядок меньше, чем у алгоритма, предложенного в [5].

Необходимые определения и понятия

Рассматриваются конечные, неориентированные графы, без петель и кратных ребер. Все неопределяемые понятия общеизвестны и их можно найти, например, в [6 - 8]. Пусть $G=(V,G)$ – граф, у которого V – множество вершин, E – множество ребер, $E \subseteq V \times V$, мощности n и m соответственно. Тройку $((v,u),v)$ будем называть инцидентором («точкой прикосновения») ребра (v,u) и вершины v. Множество таких троек обозначим I. Множество $L=V \cup E \cup I$ назовем множеством элементов графа G. Краской будем называть ресурс, который имеется у агента в неограниченном количестве, а камнем – ресурс, который имеется у агента в единичном экземпляре. Краски и камни образуют множество меток – M, которые могут быть использованы при раскраске элементов графа G. Если элемент графа метится некоторой краской, то предыдущая краска стирается. Если элемент графа метят камнем, то существующая на нем краска не уничтожается, а становится «невидимой» до тех пор, пока

камень будет находиться на элементе. Функцией раскраски графа G назовем отображение μ: L → M, где M={w, b, r, f, l}, w интерпретируется как белый цвет (краска), r – красный, b – черный, f – светло - желтый.

Простым путем называется путь, в котором все вершины u_1, u_2, … u_k попарно различны. Окрестностью O(v) вершины v будем называть множество элементов графа, состоящее из вершины v, всех вершин u смежных с v, всех ребер (v, u) и всех инциденторов ((v, u), v), ((v, u), u). Через χ обозначим цикломатическое число графа, равное (m – n + 1). Изоморфизмом графа G и графа H назовем такую биекцию φ: $V_G \rightarrow V_H$, что (u, v) ∈ E_G точно тогда, когда (φ(u), φ(v)) ∈ E_H. Изоморфные графы равны с точностью до обозначения вершин и раскраски их элементов.

Мобильный агент A передвигается по графу из вершины v в вершину u по ребру (v, u). При этом он может изменить окраску вершин v, u, ребра (v, u), инциденторов ((v, u), v), ((v, u), u) метками из множества. Находясь в вершине v, агент A воспринимает («видит») метки всех элементов окрестности O(v) и на этом основании определяет по какому ребру (v, u) он будет перемещаться и как будет перекрашивать элементы из O(v). Агент A обладает конечной, неограниченно растущей внутренней памятью, в которой фиксируется результат функционирования агента на каждом шаге, и, кроме того, постепенно выстраивается представление графа G, вначале неизвестного агенту, списками ребер и вершин. Вершину, в которой агент находится в данный момент, будем называть текущей. Помеченным путем будем назвать простой путь, образованный последовательностью вершин, помеченных камнями в порядке их посещения. Таким образом, длина помеченного пути не более чем n.

Нумерацией F: V → N вершин графа G, называется инъективное отображение множества его вершин во множество N натуральных чисел. Номер вершины v в нумерации F обозначается F(v). M - нумерацией вершин графа называется нумерация вершин графа, соответствующая порядку их обхода при поиске в глубину [7]. Древесными ребрами называются [7] ребра, порождающие дерево поиска при обходе графа в глубину, остальные ребра – обратными.

Постановка задачи

Пусть задан класс K графов: конечных, неориентированных, без петель и кратных ребер, – все элементы которых окрашены краской w. Задан агент, обладающий свойствами, описанными в пункте 2. Требуется разработать такой алгоритм функционирования (т.е. передвижения по графу и раскраски его элементов) агента A, что он, будучи помещен в произвольную вершину произвольного неизвестного графа G ∈ K, через конечное число шагов построит граф H, изоморфный G, т.е. восстановит G.

Эвристики

Напомним идею этого алгоритма [6]. Алгоритм основан на методе обход графа «в глубину». Агент идет в глубину графа в ранее не посещенные вершины до тех пор, пока это возможно. По пути агент метит инциденторы ребер меткой r (путь назад). При отсутствии возможности идти вперед, агент исследует вершину. Для этого агент метит инциденторы не посещенных ребер меткой l. После чего агент начинает отход по ребрам, инциденторы которых помечены r, с целью поиска ребер, инциденторы которых помечены l. Если же нет не посещенных ребер, то агент переходит по ребру с меткой r на инциденторе, в вершину, из которой впервые попал в текущую вершину. Если же идти назад невозможно, то агент завершает работу.

Агент обходит граф G в глубину, во время обхода он добавляет в граф H переходы по древесным ребрам. При этом, каждой вершине G ставится в соответствие (неявно) M- номер, равный количеству переходов, сделанных агентом из начальной вершины в текущую при проходе в глубину плюс количество переходов сделанных агентом при отходе назад по древесным ребрам (на данный момент времени). Переходы по обратным ребрам добавляются в граф H при исследовании вершины. После пометки инциденторов ребер исследуемой

вершины метками l, агент вычисляет M - номера всех вершин, смежных с исследуемой. Этот номер будет равен максимальному M - номеру (на данный момент) минус количество переходов назад по ребрам, инциденторы которых помечены меткой r, с момента начала процедуры исследования вершины. Таким образом, M - номер любой вершины смежной с исследуемой вершиной будет определяться однозначно.

Уменьшения длины отхода по вершинам красного пути можно добиться простыми эвристиками алгоритма из [6].

Эвристика 1. Подсчитывать число обратных ребер перед выполнением процедуры *ВОСТ*(v) и завершить ее выполнение, когда будут восстановлены все обратные ребра. Далее или переходить в конец красного пути по последнему восстановленному ребру, или возвращаться обратно по пройденным вершинам красного пути.

Эвристика 2. Если начальной вершине красного пути инцидентно только одно ребро, то после того, как она будет помечена красной краской, добавлена во множество вершин графа и будет восстановлено единственное древесное ребро, инцидентное ей, ее можно будет пометить черной краской и удалить из красного пути. Данная процедура прекращается, когда агент попадает в вершину красного пути, которой инцидентно более двух ребер. Поскольку в этом случае, при переходе по одному из них во время выполняя процедуры *ВПЕРЕД*(v), останется не восстановленным обратное ребро, для его восстановления агенту потребуется красный путь.

Первая эвристика усложняет агента из - за того, что ему необходимо подсчитывать количество обратных ребер. Вторая эвристика будет эффективна на частных случаях графов. Поэтому предлагается модификация алгоритма из [6], в которой агент использует дополнительную краску f для пометки ребра, на котором следует завершить отход по вершинам красного пути при восставнолении обратных ребер.

Модификация алгоритм восстановления графа

Алгоритм основан на методе обхода графа в глубину. Стратегия обхода графа в глубину хорошо известна [7-8]. Она изменена с учетом того, что A граф G изначально неизвестен, A может воспринимать локальную информацию о графе G и A строит на вершинах графа нумерацию в порядке посещения вершин (M - нумерация).

Предлагаемая ниже стратегия обладает рядом особенностей. Во - первых, граф G агенту не известен и агент на каждом шаге обладает информацией только о метках элементов из окрестности O(v) рабочей вершины v. Во - вторых, при прохождении вершин графа G агент создает неявную нумерацию пройденных вершин: при первом посещении вершины u она окрашивается красным цветом и ей ставится в соответствие номер по порядку. На основе этой нумерации и происходит восстановление (распознавание) графа путем построения графа H, изоморфного G. В процессе поиска агент строит неявное дерево поиска в глубину. При этом A, находясь в некоторой вершине графа метит специальной краской f не помеченные вершины и инцидентор ребра, которое в нее ведет. Относительно этого дерева все ребра разделяются: на древесные, т.е. принадлежащие этому дереву и окрашиваемые при первом прохождении по ним в красный цвет, и обратные – не принадлежащие дереву и окрашиваемые при первом прохождении в черный цвет. Древесные ребра проходятся по крайне мере 2 раза и при последнем проходе окрашиваются агентом в черный цвет. Из всех обратных ребер каждой вершины проходиться только одно обратное ребро, помеченное краской ребер f.

Древесные ребра распознаются агентом при первом проходе агента по ним. Красные вершины графа G, на каждом шаге алгоритма образуют красный путь. С помощью этого пути распознаются обратные ребра, при проходе в новую вершину красный путь удлиняется, при проходе назад – укорачивается, при восстановлении обратных ребер – не изменяется. Вершина, у которой все инцидентные ребра восстановлены, красится черной краской. Агент

завершает работу, когда красный путь становиться пустым, а все вершины черными. Таким образом, агент использует три краски (красную, черную и светло - желтую) и один камень.

Теорема 1. Выполняя алгоритм восстановления графа, агент строит представление графа H, изоморфного графу G с точностью до меток на элементах графов.

Будем предполагать, что за одну единицу времени A может выполнить все или часть следующих действий: 1) установить или снять камень; 2) изменить метку на вершине и / или инциденторе; 3) анализировать метки на элементах 1 - окрестности текущей вершины; 4) перейти из вершины в вершину по ребру, соединяющему их; 5) добавлять вершины и ребра во множества V_H и E_H.

Теорема 2. Нижняя оценка временной сложности алгоритма равна $T_*(n)= \Omega(n)$, а верхняя – $T^*(n) = O(n(t+1))$. При этом используется три краски и один камень. Здесь t – длина наибольшего простого цикла в восстанавливаемом графе.

Утверждение 1. Для модификации М2 БА верхняя оценка временной сложности равна $O(|V_3|t)$, где $V_3= \{v: deg(v) \geq 3\}$ и t – длина наибольшего простого цикла в восстанавливаемом графе.

Выводы.

Предложена модификация алгоритма из [6], для которого верхняя оценка временной сложности зависит не от высоты дерева при обходе графа в глубину, а от длины наибольшего цикла в графе, что делает оценку более точной. При этом оценка, полученная для предложенного алгоритма, не превышает оценку алгоритма из [5].

Список литературы

1. Dudek G., Jenkin M. Computational principles of mobile robotic [Text] / G. Dudek, M. Jenkin // Cambridge Univ. Press. - 2000. – 280 p.
2. Килибарда Г. Коллективы автоматов в лабиринтах [Текст] / Г. Килибарда, В. Б. Кудрявцев, Щ. Ушчумлич // Дискретная математика, 2003. Т.15, вып. 3. – С.3-40.
3. Килибарда Г. Независимые системы автоматов в лабиринтах [Текст] / Г. Килибарда, В. Б. Кудрявцев, Щ. Ушчумлич // Дискретная математика, 2003. Т.15, вып. 2. – С.3-39.
4. Килибарда Г. О поведении автоматов в лабиринтах [Текст] / Г. Килибарда, В. Б. Кудрявцев, Щ. Ушчумлич // Дискретная математика, 192. Т.4, вып. 2. – С.3-28.
5. Грунский И. С. Распознавание конечного графа блуждающим по нему агентом [Текст] / И. С. Грунский, Е. А. Татаринов // Вестник Донецкого университета. Серия А. Естественные науки, 2009. вып. 1. – С. 492-497.
6. Татаринов. Е. А. Базовый алгоритм восстановления графа [Текст] / Е. А. Татаринов // Труды ИПММ НАН Украины. – Т. 21. - 2010. – С. 216-227.
7. Касьянов В. Н. Графы в программировании: обработка, визуализация и применение [Текст] / Касьянов В. Н., Евстигнеев В. А. // СПБ.: БХВ – Петербург, 2003. – 1104 с.
8. Ахо А. Построение и анализ вычислительных алгоритмов [Текст] / А. Ахо, Дж. Хопкрофт, Дж. Ульман // М.: Мир, 1979. – 536 с.
9. Кормен Т. Алгоритмы: построение и анализ [Текст] / Т. Кормен, Ч. Лейзерсон, Р. Ривест // М.: МЦНМО, 2001. – 960 с.
10. Татаринов Е. А. М – нумерация, как метод распознавания графов [Текст] / Татаринов Е. А. // Збірник наукових праць «Питання прикладної математики математичного моделювання» -, 2010. – С. 260-272.