УДК 519.5

РАСПОЗНАВАНИЕ КОНЕЧНОГО ГРАФА БЛУЖДАЮЩИМ ПО НЕМУ АГЕНТОМ

И.С.Грунский, Е.А.Татаринов,
Институт прикладной математики и механики НАН Украины, г., Донецк

1. Введение.

Основной проблемой компьютерной науки является проблема взаимодействия управляющей и управляемой систем (управляющего автомата, агента и его операционной среды) [1, 2]. Взаимодействие этих систем, зачастую, представляется как процесс перемещения агента по помеченному графу (лабиринту) среды [3]. Целенаправленное перемещение агента в его операционной среде невозможно без формирования достаточно полной модели среды. К моделированию операционных сред определился ряд подходов, одним из которых является топологический [4]. В этом случае агенту недоступна метрическая или алгоритмическая информация о среде и доступна только информация о связях между различными областями среды. Часто такая ситуация возникает в роботике [5]. Топологические модели представляют собой графы с помеченными различными способами вершинами, дугами, инциденторами.

В настоящее время выделились три основные задачи исследования среды агентом: 1) самолокализация агента, 2) контроль соответствия модели среды и самой среды (проверка карты) и 3) построение агентом модели карты среды [5]. Задача распознавания среды (построения карты) находится в центре внимания и ей посвящено значительное количество работ (см. обзоры [6 – 8]). Известен ряд алгоритмов распознавания, получен ряд принципиальных результатов о возможности, невозможности и сложности такого распознавания с помощью тех или иных средств агента. Однако, на наш взгляд, недостаточно исследована и понята «сущность» такого распознавания.

В настоящей работе предложен новый простой алгоритм распознавания, причем, основное внимание уделено анализу того, что и как распознается на каждом шаге алгоритма. Алгоритм основан на методе обхода графа вглубь и при этом агент специальным образом окрашивает элементы графа. Алгоритм, на наш взгляд, является базовым и на его основе авторы надеются создать новые более эффективные алгоритмы, один из которых предложен в [9].

2. Основные определения и обозначения.

2.1. Раскрашенные графы.

Рассматриваются конечные, неориентированные графы без петель и кратных ребер. Все неопределяемые понятия общеизвестны и их можно найти, например, в [10-12]. Пусть $G=(V,E)$ – граф, у которого V – множество вершин, E – множество ребер, т.е. двухэлементных подмножеств (u,v), где $u,v\in V$. Тройку $((u,v),v)$ будем называть инцидентором («точкой прикосновения2) ребра (u,v) и вершины v. Множество таких троек обозначим I. Множество $L=V\cup E\cup I$ назовем множеством элементов графа G. Функцией раскраски графа G назовем отображение $\mu: L\to\{w,r,b\}$, где w интерпретируется как белый цвет (краска), r – красный, b – черный. Пара (G,μ) называется раскрашенным графом.

Последовательность $u_1,u_2,...u_k$ попарно смежных вершин называется путем в графе G, а k – длина пути. При $u_1=u_k$ этот путь называется циклом. 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 инцидентров $((v,u),v),((v,u),v)$. Мощность множеств вершин V и ребер E обозначим через n и m соответственно. Ясно что $m\le\frac{n(n-1)}{2}$. Изоморфизмом графа G и графа H назовем такую биекцию $\varphi: V_G\to V_H$, что $(u,v)\in E_G$ точно тогда, когда $(\varphi(v),\varphi(u))\in E_H$. Таким образом, изоморфные графы равны с точностью до обозначения вершин и раскраски их элементов.

2.2. Мобильный агент.

Мобильный агент A характеризуется следующими свойствами. Он 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), инцидентров $((v,u),v),((v,u),v)$. 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 v,u, (v,u), $((v,u),v),((v,u),v)$. Агент A обладает конеч-

ной, неограниченно растущей внутренней памятью, в которой фиксируется результат функционирования агента на каждом шаге, и, кроме того, постепенно выстраивается представление графа G, в начале неизвестного агенту, списками ребер и вершин.

2.3. Постановка задачи.

Требуется разработать такой алгоритм функционирования (т.е. передвижения по графу и раскраски его элементов) агента A, что будучи помещен в произвольную вершину произвольного неизвестного агенту графа G, все элементы которого окрашены цветом w, через конечное число шагов агент A восстановит граф H, изоморфный G, т.е. распознает G.

3. Распознавание графа.

3.1. Стратегия распознавания графа .

Предлагаемый алгоритм основан на стратегии поиска в глубину. Эта стратегия такова [10, 11]: идти “вглубь”, пока это возможно, и возвращаться назад, искать другой путь с еще не посещенными вершинами и не пройденными ребрами.

Стратегия поиска в глубину хорошо известна. Известен ряд алгоритмов поиска в глубину для известного графа [10 – 12]. Предлагаемая ниже стратегия обладает рядом особенностей. Во – первых, граф G агенту не известен и агент на каждом шаге обладает информацией о цветах 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 фактически ставится в соответствие номер, равный значению переменной *Сч*. 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 и 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. Обратные проходятся один раз.

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 распознавании обратного ребра – не изменяется. Вершина, у которой все инцидентные ребра распознаны, красится черным. Алгоритм оканчивает работу , когда красный путь становиться пустым, а все вершины черными.

3.2. Алгоритм распознавания графа.

Вход: граф G неизвестный агенту, все элементы графа G окрашены краской w, агент помещен в произвольную вершину v.

Выход: все элементы графа G окрашены краской b, агент находиться в вершине v; список вершин V_H и ребер E_H графа H, изоморфного G.

Данные: v – рабочая вершина графа G, в которой находиться агент , V_H, E_H – списки вершин и ребер графа H, изоморфного G (распознанного) графа. *Сч* – счетчик числа посещенных вершин графа G. $k(1)k(2)...k(t)$ – список номеров вершин красного пути, t – длина этого списка, j – счетчик, используемый для восстановления обратных ребер.

Алгоритм:

1. $Сч := 1$;
2. $t := 1$;
3. $k(t) :=$ *Сч*;
4. $V_H := \{1\}$;
5. $E_H := \varnothing$;
6. Агент красит: $\mu(v) := r$;
7. *If* $O(v)$ есть ребро (v,u), у которого $\mu(u,v) = w$ *then goto* 8 *else goto* 9;
8. *If* $O(v)$ есть ребро (v,u), у которого $\mu(u,v) = w$ и $\mu(u) = \mu(v) = r$ *then* *РАСП*(v) *else ВПЕРЕД*(v);
9. *If* в $O(v)$ есть ребро (v,u), у которого $\mu((v,u),v) = r$ *then do* НАЗАД (v); *goto* 7; *end do*
10. Агент красит: $\mu(v) := w$;
11. Печать V_H, E_H.

В строках 1 – 6 происходит инициализация алгоритма: вершина v, посещенная впервые красится в красный цвет, ей неявно присваивается номер 1, т.о. создается красный путь длины 1 и список номеров красного пути $k(1)$. Формируются также начальные значения множеств путей и ребер графа H.

В строках 7 – 9 производится анализ окрестности $O(v)$ рабочей вершины v и выбор одной из трех процедур. Если в окрестности $O(v)$ есть еще не пройденные ребра, то в строке 8 в случае, когда есть обратное не пройденное ребро, выполняется процедура *РАСП*(v), если такого ребра нет, то выполняется процедура *ВПЕРЕД*(v). Если в $O(v)$ все ребра уже пройдены и есть ребро из красного пути, то выполняется процедура *НАЗАД*(v). Если такого ребра нет, то агент красит $\mu(v) := b$, (строка 10), печатается граф (строка 11) и оканчивает работу. Подчеркнем, что после выполнения вышеуказанных трех процедур осуществляется переход к строке 7.

Рассмотрим процедуры *ВПЕРЕД*(v), *НАЗАД*(v), *РАСП*(v)

ВПЕРЕД(v)

1. Агент A выбирает из $O(v)$ произвольное ребро (v,u), у которого $\mu(u) = \mu(v,u) = w$, переходит по нему в вершину u;
2. Агент красит: $\mu(v,u) := r$, $\mu(u) := r$, $\mu((v,u),u) := r$;
3. *Сч* := *Сч*+1;
4. $k(t+1) := Сч$:
5. $t := t+1$;
6. $v := u$;
7. $V_H := V_H \cup \{Сч\}$;
8. $E_H := E_H \cup \{k(t-1), k(t)\}$;

В строке 1 агент впервые проходит по белому ребру (v,u) и впервые посещает белую вершину u. В строке 2 он красит $u, (v,u)$, и инцидентор $((v,u),u)$ красным цветом, т.е. удлиняет красный путь. В строках 3 – 5 удлиняется список номеров вершин красного пути. В строках 7 – 8 создается новая вершина графа H и распознается новое древесное ребро графа H. После этого происходит возврат в строку 7 алгоритма распознавания.

НАЗАД(v)

1. Агент A выбирает ребро $(v,u) \in O(v)$, у которого $\mu(v,u) = r$, и $\mu((v,u),v) = r$, переходит по нему в вершину u.
2. Агент красит: $\mu(v) := b, \mu(v,u) := b$;
3. $v := u$;
4. $t := t-1$;
5. Из списка $k(1)...k(t+1)$ удаляется элемент $k(t+1)$;

В строке 1 агент проходит по красному ребру (v,u) поскольку все остальные ребра из $O(v)$ уже черные. В строке 2 это ребро красится черным, вершина v краситься черным и тем самым удаляется из красного пути. Происходит укорочение этого пути на одну вершину. В строках 4 – 5 происходит соответствующее укорачивание списка $k(1)...k(t)$ номеров вершин красного пути. После этого происходит возврат в строку 7 алгоритма распознавания.

РАСП(v)

1. Агент выбирает из $O(v)$ произвольное ребро (v,u), у которого $\mu(v) = \mu(u) = r$ и $\mu(v,u) = w$ и переходит по нему в вершину u;
2. Агент красит: $\mu(v,u) := b$;
3. Агент выбирает из $O(u)$ ребро (u,z), у которого $\mu(u,z) = r$ и $\mu((u,z),z) = r$, и переходит по нему в вершину z;
4. $u := z$;
5. $j := 1$;
6. *while* в окрестности $O(u)$ есть ребро (u,z), у которого $\mu(u,z) = r$ и $\mu((u,z),z) = r$ *do*
7. агент переходит по ребру (v,u) в вершину z;

8. $u := z$;
9. $j := j + 1$;
end do
11. $E_H := E_H \cup \{(k(t), k(t-j))\}$;

В строке 1 агент выбирает еще не пройденное обратное (белое) ребро (v,u) с красными вершинами u, v и переходит по нему в вершину u. В строке 2 агент красит это ребро черным цветом. После этого агент оказывается в вершине u, принадлежащей красному пути. В строках 3 – 10 идет по красному пути из вершины с меньшим номером в вершину с большим номером (в неявной нумерации), до тех пор пока не попадет в исходную вершину v. Заметим, что если в $O(v)$ есть несколько обратных ребер, то в силу строк 7 – 9 алгоритма распознавания они будут распознаны все и только после этого агент пойдет из v вперед или назад. С помощью счетчика j формируется новое ребро $(k(t), k(t-j))$ графа H, распознанное при выполнении этой процедуры. После этого происходит возврат в строку 7 алгоритма распознавания.

4. Свойства алгоритма распознавания

В начале алгоритма, если $n \geq 3$, то не меньше 2 – х раз выполняется процедура *ВПЕРЕД*(v), поскольку в рассматриваемых графах простых циклов длины меньшей 3 нет. При выполнении этой процедуры агент посещает новую вершину графа G и создает новую вершину графа H. Следовательно, в процессе выполнения алгоритма устанавливается неявная нумерация $\varphi : V_G \to V_H$ вершин графа G в вершины графа H, где равенство $\varphi(v) = t$ устанавливается, когда вершина v красится в красный цвет и $t = Сч$. Эта нумерация является биекцией, поскольку в связном графе G все вершины достижимы из начальной и поэтому все вершины посещаются агентом, т.е. красятся в красный цвет.

Поскольку алгоритм основан на стратегии обхода графа вглубь, то все ребра графа G являются либо древесными (при первом прохождении красятся в красный цвет), либо обратными (и при единственном прохождении красятся в черный цвет). Это утверждение является простым следствием теоремы 23.9 книги [11] (см. стр. 452). Из описания алгоритма следует, что агент проходит все ребра графа G, поскольку при окончании алгоритма все ребра становятся черными. При выполнении процедуры *ВПЕРЕД*(v) агент распознает древесное ребро (v,u) и так нумерует вершину u, что ребру (v,u) однозначно соответствует ребро $(\varphi(v), \varphi(u))$ графа H. При выполнении процедуры *РАСП*(v) агент распознает обратное ребро (v,u) графа G и устанавливает ему в однозначное соответствие ребро $(\varphi(v), \varphi(u))$ графа H. Следовательно, φ является изоморфизмом графа G на граф H.

Теорема 1 Выполняя алгоритм распознавания, агент распознает любой граф G с точностью до изоморфизма.

Подсчитаем временную и емкостную сложность в равномерной шкале [10]. Для этого рассмотрим свойства красного пути. Из описания алгоритма следует, что на каждом шаге алгоритма красный путь – это простой путь, соединяющий начальную вершину v с номером $\varphi(v) = 1$ с вершиной u с номером $\varphi(u) = Сч$. Следовательно, длина красного пути не превосходит n. При выполнении процедуры *ВПЕРЕД*(v) агент проходит одно ребро, при выполнении процедуры *НАЗАД*(v) – то же одно ребро, при выполнении процедура *РАСП*(v) агент проходит одно обратное ребро и не более $n-1$ ребер красного пути. При выполнении процедуры *РАСП*(v) агент проходит фактически цикл, состоящий из обратного ребра и некоторого конечного отрезка красного пути, соединяющего вершины инцидентные обратному ребру.

При выполнении алгоритма агент проходит дерево поиска в глубину, состоящее из $n-1$ ребер, и $m-n+1$ обратных ребер. При подсчете временной сложности алгоритма, будем считать, что инициализация алгоритма (строки 1 – 6 описания алгоритма распознавания) анализ окрестности $O(v)$ рабочей вершины и выбор одной из трех процедур (строки 7 – 9) занимают некоторое постоянное число единиц времени. Так же будет считать, что выбор ребра при выполнении процедур и проход по ребру осуществляется за 1 единицу времени. Тогда временная сложность алгоритма определяется следующими соотношениями:

1. Инициализация выполняется один раз и ее асимптотическая сложность равна $O(1)$;

2. процедура вперед выполняется равно $n-1$ раз и общее время ее выполнения оценивается как $O(n)$;
3. аналогично общее время выполнения процедуры *НАЗАД*(v) оценивается как $O(n)$;
4. выполнение процедуры *РАСП*(v) один раз оценивается как $O(n)(m-n+1)$, т.е. как $O(n^3)$;

Следовательно, суммарная временная сложность $T(n)$ алгоритма удовлетворяет соотношению: $T(n)=O(n^3)$.

Емкостная сложность $S(n)$ алгоритма определяется сложностью списков $V_H, E_H, k(1)...k(t)$, сложность которых соответственно определяется величинами $O(n), O(n^2), O(n)$. Следовательно: $S(n)=O(n^2)$.

Теорема 2 Временная сложность алгоритма распознавания равна $O(n^3)$, а емкостная – $O(n^2)$. При этом алгоритм использует 2 краски.

5. Выводы.

В работе предложен новый алгоритм распознавания графа. Выполняя алгоритм распознавания, агент A строит граф H изоморфный исследуемому графу G. При этом агенту требуется 2 краски. Алгоритм останавливается через конечное число шагов, число которых является величиной $O(n^3)$. При этом емкостная сложность есть величина $O(n^2)$. Предложенный алгоритм требует меньшее количество красок, чем известные [9]. Однако имеет худшую временную сложность.

РЕЗЮМЕ

Розглядається задача розпізнавання графу мобільним агентом. Агент рухається графом, змінює та считує помітки на елементах графа, оперує з внутрішньою пам'ятью, в якій він будує представлениіє досліджуваного графа. Для розпізнавання агент потребує 2 різні кольори, квадратичну пам'ять та кубічне (від числа вершин у графі) число кроків. Метод базується на методі обходу графа в глибину. При цьому враховується, що на початку досліджуваний граф не відомий агенту.

SUMMARY

Graphs recognition problem by means an agent is considered. The agent moves on unknown graph, changes and reads marks of graph elements, operates it's internal memory, in which it builds explored graph representations. For exploring any finite not oriented graph agent needs two different marks, quadratic memory and cubic numbers (by graph vertices number) of steps. The method based on depth-first traversal method. In this connection account what agent know about graph nothing.

СПИСОК ЛИТЕРАТУРЫ

1. Глушков В.М., Летичевский А.А. Теория дискретных преобразователей // Избранные вопросы алгебры и логики. – Новосибирск: Наука, 1973. – с. 5 – 20.
2. Капитонова Ю.В., Летичевский А.А. Математическая теория проектирования вычислительных систем. – М.: Наука, 1988. – 296 с.
3. Кудрявцев В.Б., Алешин С.В., Подкозлин А.С. Введение в теорию автоматов. М.: Наука, 1985. – 320 с.
4. Kuipers B. The spatial semantic hierarchy // Artificial Intelligence. – 2000. – v.119, № 1 – 2. – p. 191 – 233.
5. Dudek G, Jenkin M. Computational principles of mobile robotics – Cambridge Univ. press, Cambridge, 2000. – 280 p.
6. Калибарда Г., Кудрявцев В. Б., Ущумлич Ш. Независимые системы автоматов в лабиринтах // Дискретная математика – 2003. – т. 15, вып. 2. – с. 3 – 39.
7. Калибарда Г., Кудрявцев В. Б., Ущумлич Ш. Независимые системы автоматов в лабиринтах // Дискретная математика – 2003. – т. 15, вып. 3. – С. 3 – 39.
8. Fraigniaud P., Jecincas D., Peer G., Pelc A., Peleg D. Graph Exploration by a finite automaton // proc. 29 th Internac. Symp. On Math. Foundation of Computer Science (MFCS), LNCS 3153- 2004. – p. 451 – 462.
9. Грунский И. С., Татаринов Е. А. Алгоритм распознавания графов // Тр. 4 междунар. Конф. «Параллельные вычисления и задачи управления» PACO'2008, Москва, Ин-т Проблем Управления им. В. А, Трапезникова РАН, 2008 – с. 1483 – 1498.
10. Ахо А., Хопкрофт Дж., Ульман Дж. Построение и анализ вычислительных алгоритмов. – М.: Мир, 1979. – 536 с.
11. Кормен Т., Лейзерсон Ч., Ривест Р. Алгоритмы: построение и анализ. – М.: МЦНМО, 2001. – 960 с.
12. Касьянов В. Н., Евстигнеев В. А. Графы в программировании: обработка, визуализация и применение. – СПБ.: БХВ – Петербург, 2003. – 1104 с.

Надійшла до редакції 25.02.2009 р.