КОМПЬЮТЕРНЫЕ ИССЛЕДОВАНИЯ
И МОДЕЛИРОВАНИЕ 2013 Т. 5 № 4 С. 525–532

Ки&М

МАТЕМАТИЧЕСКИЕ ОСНОВЫ И ЧИСЛЕННЫЕ МЕТОДЫ МОДЕЛИРОВАНИЯ

УДК: 519.17

Использование коллектива агентов для распознавания графа

А. В. Стёпкин

Институт прикладной математики и механики Национальной академии наук Украины,
Украина, 83114, г. Донецк, ул. Розы Люксембург, д. 74

E-mail: stepkin.andrey@rambler.ru

Получено 13 июня 2013 г.

В работе рассматривается задача распознавания графов коллективом агентов. Два агента-исследователя одновременно передвигаются по графу, считывают и изменяют метки элементов графа, передают необходимую информацию агенту-экспериментатору, который строит представление исследуемого графа. Построен алгоритм распознавания линейной (от числа вершин графа) временной сложности, квадратичной емкостной сложности и коммуникационной сложности равной $O\left(n^2 \cdot \log(n)\right)$, где n — число вершин графа. Для распознавания два, передвигающиеся по графу, агента используют по две различные краски (всего три краски). Алгоритм основан на методе обхода графа в глубину.

Ключевые слова: распознавание графа, коллектив агентов

Using collective of agents for exploration of graph

A. Stepkin

Institute of applied mathematics and mechanics of the National academy of science of Ukraine, 74 Roza Luksemburg street, Donetsk, 83114, Ukraine

Abstract. — Problem of exploration finite undirected graphs by a collective of agents is considered in this work. Two agents-researchers simultaneously move on graph, they read and change marks of graph elements, transfer the information to the agent-experimenter (it builds explored graph representation). It was constructed an algorithm linear (from amount of the graph's nodes) time complexity, quadratic space complexity and communication complexity, that is equal to $O\left(n^2 \cdot \log(n)\right)$. Two agents (which move on graph) need two different colors (in total three colors) for graph exploration. An algorithm is based on depth-first traversal method.

Keywords: graph exploration, collective of agents

Citation: *Computer Research and Modeling*, 2013, vol. 5, no. 4, pp. 525–532 (Russian).

Введение

В настоящее время в мире существует огромное количество различного рода сред, требующих изучения [Albers, Henzinger, 2000; Deng, Papadimitriou, 1999] (начиная с исследования среды при помощи наноботов и заканчивая исследованием поверхностей отдаленных планет с помощью планетоходов). Это является одной из причин активного развития такого направления математической кибернетики, как теория дискретных динамических систем. Исследование среды — это своего рода дискретная система, представленная как модель взаимодействия управляющей и управляемой систем (управляющего автомата и операционной среды), взаимодействие которых зачастую представляется как процесс перемещения управляющего автомата по графу управляемой системы [Килибарда, Кудрявцев, Ушчумлич, 2003a]. Что в конечном итоге и привело к обширному и интенсивно развивающемуся исследованию поведения автоматов в лабиринтах [Килибарда, Кудрявцев, Ушчумлич, 2003b; Dudek et al., 1993; Грунский, Сапунов, 2002].

Исследованию графа при помощи одного агента посвящено много работ, получено множество результатов о возможности и сложности такого распознавания. При этом, на наш взгляд, остается малоисследованно распознавание графа при помощи нескольких блуждающих по нему агентов. Что делает актуальной задачу проведения систематического исследования экспериментов по распознаванию графа несколькими блуждающими по нему агентами, то есть создания маршрутов движения агентов по неизвестному графу, разметки его элементов, сбора и обработки локальной информации о графе и способов построения графа по этой информации, с точностью до отметок на элементах графа. А также задачи, связанные с оптимизацией расходов ресурсов и затрат времени.

При распознавании графа несколькими блуждающими по нему агентами основной проблемой является проблема эффективности их взаимодействия с целью уменьшения затрат времени и памяти на распознавание. Требуется разработать такие алгоритмы движения, при которых блуждающие агенты не мешают друг другу и не дублируют работу друг друга. Это предполагается достичь за счет прямой связи между агентами. В данной работе рассматривается коллектив из трех агентов: два агента-исследователя блуждают по графу и перекрашивают его элементы, передавая полученную информацию агенту-экспериментатору. Взаимодействие агентов-исследователей осуществляется за счет окраски элементов графа.

Ранее полученные нами алгоритмы [Грунский, Стёпкин, 2009; Стёпкин, 2012] решения рассматриваемой задачи имеют кубическую (от числа вершин графа) и квадратическую временные сложности соответственно при неизменной квадратической емкостной сложности. Рассмотрим алгоритм решения нашей проблемы в случае, когда два агента-исследователя (АИ) A и B одновременно передвигаются по неизвестному конечному неориентированному графу без петель и кратных ребер, обмениваются необходимой информацией с агентом-экспериментатором (АЭ), который и восстанавливает исследуемый граф. В работе предложен алгоритм построения маршрутов АИ по графу, позволяющих АЭ точно восстановить граф среды. Каждому АИ для работы требуется две краски: у A это r и b, у B — y и b. Алгоритм основан на методе обхода графа в глубину [Кормен, Лейзерсон, Ривест, 2001], имеет линейную временную сложность и квадратическую емкостную сложность. Из работы [Стёпкин, 2012] следует, что временная и коммуникационная сложности алгоритма равны $O(n^2)$, а в предлагаемом алгоритме, за счет увеличения коммуникационной сложности в $\log(n)$ раз, достигнуто снижение временной сложности в n раз. При описании алгоритма используются результаты и обозначения из [Грунский, Стёпкин, 2009; Стёпкин, 2012].

Основные определения и обозначения

Пусть $G = (V, E)$ — связный неориентированный конечный граф без петель и кратных ребер, где V — множество вершин, E — множество ребер (двухэлементных подмножеств (v, u), где $v, u \in V$). Тройку $((v, u), u)$ будем называть инцидентором ребра (v, u) и вершины u. Множество таких троек обозначим I. Множество $L = V \cup E \cup I$ назовем множеством элементов графа G. Сюръективное отображение $\mu : L \to \{w, r, y, ry, b\}$, где w интерпретируется как белый цвет, r — красный, y — желтый, ry — красно-желтый, b — черный, назовем функцией раскраски графа G. Пара (G, μ) называется раскрашенным графом. Последовательность $u_1, u_2, ..., u_k$ попарно смежных вершин графа G называется путем длины k. Окрестностью $Q(v)$ вершины v будем называть множество элементов графа, состоящее из вершины v, всех вершин u смежных с v, всех ребер (v, u) и всех инцидентров $((v, u), v), ((v, u), u)$. Мощность множеств вершин V и ребер E обозначим через n и m соответственно. Ясно что $m \leq \frac{n(n-1)}{2}$. Изоморфизмом графа G и графа H назовем такую биекцию $\varphi : V_G \to V_H$, что $(v, u) \in E_G$ точно тогда, когда $(\varphi(v), \varphi(u)) \in E_H$. Таким образом, изоморфные графы равны с точностью до обозначения вершин и раскраски их элементов.

Коллектив агентов, распознающий ранее им неизвестный граф G, состоит из трех агентов: два АИ A и B и один АЭ. АИ перемещаются по графу и могут изменять окраску элементов графа. АЭ на основании сообщений от АИ создает представление графа. Мобильные агенты A и B имеют конечную на каждом шаге, но растущую память. В начале работы АИ A и B помещаются в произвольные несовпадающие вершины графа G, нумеруют их и передают номера АЭ, который помещает их в множество вершин V_H. Агенты передвигаются по графу из вершины v в вершину u по ребру (v, u), могут изменять окраску вершин v, u, ребер (v, u), инцидентров $((v, u), v), ((v, u), u)$, а также записывают в вершины номера. Находясь в вершине v, АИ воспринимает метки всех элементов окрестности $Q(v)$ и номера смежных с ней вершин, на основании этой информации определяет, по какому ребру будет дальше перемещаться, и как будет окрашивать элементы графа. АЭ передает, принимает и идентифицирует сообщения, полученные от АИ, обладает конечной, неограниченно растущей внутренней памятью, в которой фиксируется результат функционирования АИ на каждом шаге и, кроме того, строится представление графа G, вначале неизвестного агентам, списками ребер и вершин.

Алгоритм работы агентов

Рассмотрим подробно режимы работы АИ. При описании режимов в скобках указываются сообщения, которые будут отправлены агентами-исследователями агенту-экспериментатору, попав в рассматриваемую ситуацию («СООБЩ_АИ_А»; «СООБЩ_АИ_В»). АЭ, в свою очередь, обрабатывает полученное сообщение и отправляет АИ данные необходимые для завершения хода.

Обычный режим работы (ОРР). АИ выбирает в окрестности вершины, в которой он находится, произвольную белую вершину и переходит в нее. Таким образом АИ движется вперед по белым вершинам, окрашивая вершины, соединяющие их ребра и дальние инциденторы в «свой» цвет («ВПЕРЕД_А»; «ВПЕРЕД_В»), получив от АЭ номера, записывает их в вершины. Если в окрестности вершины, в которой стоит АИ не найдется белой вершины, для дальнейшего движения вперед, то АИ A отправляет АЭ сообщение «ПРИСВ_Amr», на что затрачивает один ход, далее возвращается назад, окрашивая пройденные вершины, ребра и ближние инциденторы в черный цвет («НАЗАД_А»). АИ B, не обнаружив белой вершины для дальнейшего движения вперед, сразу начинает движение назад, окрашивая пройденные вершины, ребра и ближние инциденторы в черный цвет («НАЗАД_В»).

Вернувшись в начальную вершину, АИ завершает работу («СТОП_А»; «СТОП_В»). Таким образом строятся два дерева методом обхода в глубину.

Режим распознавания обратных ребер (РРОР). Обратное ребро — это белое ребро, дальняя вершина которого окрашена в «свой» цвет. Если при движении вперед было обнаружено обратное ребро, то АИ сканирует окрестность вершины в которой находится, считывает номера всех смежных вершин, инцидентных обратным ребрам, и отправляет список номеров АЭ («ОБР_А($x_1, x_2, ..., x_l$)»; «ОБР_В($k_1, k_2, ..., k_l$)», где x_i, k_i — номера, записанные агентами A и B соответственно в вершинах своего пути).

Режим распознавания перешейков (РРП). Под перешейком подразумевается ребро, соединяющее вершины, принадлежащие областям работы разных агентов. Этот режим немного отличается для каждого из АИ. Если при движении вперед агент A обнаружил в вершине перешейки и ни в одной из дальних вершин этих перешейков нет агента B (или агент B находится в одной из этих вершин, но уже распознал все ранее обнаруженные ним перешейки в эту вершину, или же B выполняет возврат назад по своему пути), то агент A отправляет АЭ все номера дальних вершин перешейков («ПЕР_А($x_1, x_2, ..., x_l$)», где x_i — номера, записанные агентом B, в вершинах своего пути). Если же агент B находится в одной из дальних вершин перешейков и еще не распознавал инцидентные ей перешейки или не возвращается назад по своему пути, то агент A отправляет АЭ номера всех дальних вершин обнаруженных перешейков, кроме номера вершины, в которой находится агент B. Заметим, что агент A узнает о нахождении B в смежной вершине в результате сканирования окрестности на наличие перешейков, но о том, можно ли распознавать перешеек, в дальней вершине которого находится B, агент A узнает из значения переменной *mr_A*, запрашиваемой у АЭ. Если при движении вперед перешейки обнаружил агент B и ни в одной из дальних вершин этих перешейков нет агента A (или же A находится в одной из этих вершин, но выполняет возврат назад по своему пути), то B передает АЭ номера всех дальних вершин обнаруженных перешейков («ПЕР_В($k_1, k_2, ..., k_l$)», где k_i — номера, записанные агентом A, в вершинах своего пути). Если же агент A находится в одной из дальних вершин перешейков и не выполняет возврат назад по своему пути, то B не выполняет никаких действий до ухода A из окрестности вершины, в которой находится B. Агент B узнает о нахождении A в смежной вершине в результате сканирования окрестности на наличие перешейков, но о том можно ли распознавать перешейки при наличии в дальней вершине одного из них агента A, агент B узнает из значения переменной *mr_B*, запрашиваемой у АЭ.

При одновременном попадании двух АИ в одну белую вершину каждый АИ окрашивает вершину наполовину, и она становится красно-желтой. Агент B на следующем шаге отступает назад по своему пути, удаляя метки, оставленные им на предыдущем шаге (удаляется краска с ребра и ближнего инцидентора), и переключается в ОРР. Агент A видит разноцветную вершину как свою, но при распознавании окрашивает в черный цвет всю вершину.

При попадании АИ в ситуацию, когда в вершине возможен выбор сразу нескольких режимов работы, то первым будет выбран РРП, за ним РРОР и наконец ОРР. Попадание двух АИ в одну белую вершину в этом списке не рассматривается, так как такая ситуация приведет к изменениям в работе исключительно агента B, и в этот момент другие режимы работы для него будут недоступны.

Выполняя обход графа, агенты A и B создают соответственно красный и желтый пути. Рассмотрим принцип построения агентами пути «своего» цвета. При движении в белую вершину красный (желтый) путь удлиняется, при движении назад по своему пути — укорачивается. Если АИ вернулся в вершину, из которой начал обход графа, а в ее окрестности не оказалось белых вершин, то АИ окрашивает эту вершину в черный цвет. Алгоритм заканчивает работу, когда красный и желтый пути становятся пустыми, а все вершины черными.

Выполняя обход графа G, агенты создают нумерацию посещенных вершин. Первый раз посетив вершину, агент A окрашивает ее в красный цвет (агент B — в желтый цвет), записывает в память вершины соответствующий номер (полученный от АЭ), равный значению переменной

Сч_А (*Сч_В* для агента *B*). Восстановление графа *G* происходит на основе созданной агентами-исследователями нумерации путем построения графа *H* изоморфного *G*.

Рассмотрим алгоритм работы АЭ:

Вход: списки сообщений *M* и *N* от АИ.

Выход: список вершин V_H и ребер E_H графа *H*, изоморфного графу *G*.

Данные: V_H, E_H — списки вершин и ребер графа *H*, изоморфного графу *G*. *Сч_А*, *Сч_В* — счетчики числа посещенных вершин агентами *A* и *B* соответственно. $STOP\_A, STOP\_B$ — переменные, используемые агентами *A* и *B* соответственно для сигнализации АЭ о завершении распознавания своей области. mr_A — переменная, которая принимает значения «1» или «0». Значение «1» позволяет агенту *A* распознавать перешейки, в том числе и те, во второй вершине которых стоит агент *B*. Значение «0» позволяет агенту *A* распознавать только те перешейки (если они существуют), во второй вершине которых нет агента *B*. mr_B — переменная, которая принимает значения «1» или «0». Значение «1» позволяет агенту *B* распознавать перешейки даже при наличии в дальней вершине одного из перешейков агента *A*. Значение «0» запрещает агенту *B* распознавание перешейков, если в дальней вершине одного из перешейков находится агент *A*. $r(1), r(2), ..., r(t)$ — список номеров вершин красного пути, где t — длинна этого списка. $y(1), y(2), ..., y(p)$ — список номеров вершин желтого пути, где p — длинна этого списка. *Mes* — текущее сообщение.

1. *Сч_А*:= 1, *Сч_В*:= 1, $M := \varnothing, N := \varnothing, E_H := \varnothing, STOP\_A := 0,$
   $STOP\_B := 0, mr\_A := 0, mr\_B := 0, t := 1, p := 1, r(t) :=$ *Сч_А*,
   $y(p) :=$ *Сч_В*, $V_H := \{A[1], B[1]\}$;
2. *while* $(STOP\_A = 0)$ *or* $(STOP\_B = 0)$ *do*
3. *if* $M \neq \varnothing$ *then do*
4. прочитать в *Mes* сообщение и удалить его из очереди *M*;
5. *ОБР_СП_А*();
6. *end do*;
7. *if* $N \neq \varnothing$ *then do*
8. прочитать в *Mes* сообщение и удалить его из очереди *N*;
9. *ОБР_СП_В*();
10. *end do*;
11. *end do*;
12. печать V_H, E_H. Рассмотрим процедуры, используемые в алгоритме.

ОБР_СП_А():

1. *if Mes* = «ПЕР_А($x_1, x_2, ..., x_l$)» *then* ПЕР_А($x_1, x_2, ..., x_l$);
2. *if Mes* = «ОБР_А($x_1, x_2, ..., x_l$)» *then* ОБР_А($x_1, x_2, ..., x_l$);
3. *if Mes* = «ВПЕРЕД_А» *then* ВПЕРЕД_А();
4. *if Mes* = «ПРИСВ_Amr» *then* ПРИСВ_Amr();
5. *if Mes* = «НАЗАД_А» *then* НАЗАД_А();
6. *if Mes* = «СТОП_А» *then* СТОП_А().

ПЕР_А($x_1, x_2, ..., x_l$): выполняется операция:
$E_H := E_H \cup \{(A[r(t)], B[x_1]); (A[r(t)], B[x_2]); ...; (A[r(t)], B[x_l])\}$.

ОБР_А($x_1, x_2, ..., x_l$): выполняется операция:
$E_H := E_H \cup \{(A[r(t)], A[x_1]); (A[r(t)], A[x_2]); ...; (A[r(t)], A[x_l])\}$.

ВПЕРЕД_А(): выполняются операции: *Сч_А*:=*Сч_А*+1; $t := t + 1$.
$r(t) :=$ *Сч_А*; $V_H := V_H \cup \{A[$*Сч_А*$]\}$; $E_H := E_H \cup \{(A[r(t-1)], A[r(t)])\}$; $mr\_B := 0$.

ПРИСВ_Amr(): $mr\_B := 1$.

НАЗАД_А(): из списка $r(1), ..., r(t)$ удаляется элемент $r(t)$; $t := t - 1$.

СТОП_А(): $STOP\_A := 1$.

Процедуры работы со списком сообщений от агента B, которые не рассмотрены ниже, аналогичны процедурам работы со списком сообщений от агента A.

ОБР_СП_B():

1. *if* Mes = «ВОЗВРАТ_B» *then* ВОЗВРАТ_B();
2. *if* Mes = «ПЕР_B($k_1, k_2, ..., k_l$)» *then* ПЕР_B($k_1, k_2, ..., k_l$);
3. *if* Mes = «ОБР_B($k_1, k_2, ..., k_l$)» *then* ОБР_B($k_1, k_2, ..., k_l$);
4. *if* Mes = «ВПЕРЕД_B» *then* ВПЕРЕД_B();
5. *if* Mes = «НАЗАД_B» *then* НАЗАД_B();
6. *if* Mes = «СТОП_B» *then* СТОП_B().

ВОЗВРАТ_B(): $E_H := E_H \backslash \{(y(p-1), y(p))\}$; $V_H := V_H \backslash \{B[Сч\_B]\}$;
$Сч\_B := Сч\_B - 1$; $p := p - 1$; $y(p) := Сч\_B$.

ПЕР_B($k_1, k_2, ..., k_l$): выполняются операции:
$E_H := E_H \cup \{(B[y(p)], A[k_1]); (B[y(p)], A[k_2]); ...; (B[y(p)], A[k_l])\}$;
$mr\_A := 1$.

ВПЕРЕД_B(): выполняются операции: $Сч\_B := Сч\_B + 1$; $p := p + 1$;
$y(p) := Сч\_B$; $V_H := V_H \cup \{B[Сч\_B]\}$; $E_H := E_H \cup \{(B[y(p-1)], B[y(p)])\}$; $mr\_A := 0$.

НАЗАД_B(): из списка $y(1), ..., y(p)$ удаляется элемент $y(p)$; $p := p - 1$; $mr\_A := 1$.

Свойства алгоритма распознавания

Процедуры ВПЕРЕД_A() и ВПЕРЕД_B() выполняются АЭ во время посещения агентами-исследователями белых вершин исследуемого графа G. Процедурами АЭ ВПЕРЕД_A() и ВПЕРЕД_B() создается по одной новой вершине графа H. При одновременном попадании агентов A и B в одну белую вершину процедурами ВПЕРЕД_A() и ВПЕРЕД_B() будут созданы две новые вершины графа H. Для того чтобы не допустить подобное дублирование вершин, на следующем шаге агент B процедурой ВОЗВРАТ_B() удалит вершину, созданную им на предыдущем шаге. Таким образом, процесс выполнения описанного алгоритма индуцирует отображение $\varphi : V_G \to V_H$. Причем $\varphi(v) = t$ (когда вершина v окрашена в красный цвет и $t = Сч\_A$) и $\varphi(s) = p$ (когда вершина s окрашена в желтый цвет и $p = Сч\_B$). Указанное отображение φ является биекцией, поскольку в связном графе G все вершины достижимы из начальных вершин.

При выполнении процедуры ВПЕРЕД_A() или ВПЕРЕД_B() АЭ 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 процедур *ОБР_A*() или *ОБР_B*() АЭ распознает обратные ребра (v, u) графа G и ставит им в однозначное соответствие ребра $(\varphi(v), \varphi(u))$ графа H. При выполнении процедур *ПЕР_A*() или *ПЕР_B*() АЭ распознает перешейки (v, u) графа G и ставит им в однозначное соответствие ребра $(\varphi(v), \varphi(u))$ графа H. Следовательно, φ является изоморфизмом графа G на граф H.

Теорема. *Три агента, выполнив алгоритм распознавания на графе G, распознают рассматриваемый граф с точностью до изоморфизма.*

Подсчитаем временную, емкостную и коммуникационную сложности алгоритма в равномерной шкале [Кормен, Лейзерсон, Ривест, 2001]. Из описания алгоритма следует, что на каждом шаге алгоритма красный (желтый) путь — это простой путь, соединяющий начальную вершину v (s — в случае агента B) с номером $\varphi(v) = 1$ ($\varphi(s) = 1$) с вершиной u (z) с номером $\varphi(u) = Сч\_A$ ($\varphi(z) = Сч\_B$). Следовательно, общая длина красного и желтого пути не превосходит n.

При однократном выполнении процедур из ОРР АИ проходит одно ребро либо же, не выполняя передвижений, отправляет одно сообщение, на что уходит один ход. При однократном выполнении процедур из РРОР АИ распознают не более $n-2$ обратных ребер, на что затрачивают

один ход. При однократном выполнении процедуры из РРП АИ распознают не более $n-2$ перешейка, на что так же уходит один ход. При одновременном попадании двух АИ в одну белую вершину, агент A не меняет режим работы, а агент B затрачивает один ход на возврат в свою область работы. При подсчете временной сложности алгоритма будем считать, что инициализация алгоритма, анализ окрестности $Q(v)$ рабочей вершины и выбор одной из возможных процедур занимают некоторое постоянное число единиц времени. Так же будем считать, что выбор ребер, проход по ним АИ и обработка сообщений отправленных на данном этапе осуществляется за 1 единицу времени. Тогда временная сложность алгоритма определяется следующим образом. Процедуры ОРР выполняются не более чем $3\cdot(n-2)+2$ раз, общее время их выполнения оценивается как $O(n)$. Процедуры РРОР выполняются не более чем $n-2$ раз, то есть общее время их выполнения оценивается как $O(n)$. Процедуры РРП выполняются не более чем $n-2$ раз, то есть общее время их выполнения оценивается как $O(n)$. Время простоя агентов в ожидании оценивается как $O(n)$. Следовательно, суммарная временная сложность $T(n)$ алгоритма удовлетворяет соотношению $T(n)=O(n)$.

Емкостная сложность $S(n)$ алгоритма определяется сложностью списков $V_H, E_H, r(1)...r(t), y(1)...y(p)$, сложность которых соответственно определяется величинами $O(n\cdot\log(n)), O\left(n^2\right), O(n\cdot\log(n)), O(n\cdot\log(n))$. Следовательно, $S(n)=O\left(n^2\right)$.

Коммуникационная сложность $K(n)$ алгоритма определяется объемом информации, которой необходимо обменяться агентам, для распознавания графа. При работе агентов в ОРР объем передаваемой АИ информации оценивается как $O(n)$. АЭ при этом передаст объем информации, оцениваемый как $O(n\cdot\log(n))$. При работе агентов в РРОР и в РРП объем переданной информации оценивается как $2\times O\left(n^2\cdot\log(n)\right)$. Следовательно, суммарная коммуникационная сложность $K(n)$ алгоритма удовлетворяет соотношению $K(n)=O\left(n^2\cdot\log(n)\right)$.

Теорема. *Временная сложность алгоритма распознавания равна $O(n)$, емкостная — $O\left(n^2\right)$, а коммуникационная — $O\left(n^2\cdot\log(n)\right)$. При этом алгоритм использует 3 краски.*

Выводы

В работе предложен новый алгоритм распознавания конечных неориентированных графов временной сложности $O(n)$, емкостной сложности $O\left(n^2\right)$ и коммуникационной сложности $O\left(n^2\cdot\log(n)\right)$. АИ имеют конечную на каждом шаге, но растущую память и используют по две краски каждый (всего три краски).

Автор выражает глубокую признательность своему научному руководителю проф. каф. ПОИС ф-та компьютерных наук и технологий И. С. Грунскому за оказанную помощь в работе.

Список литературы

Грунский И. С., Сапунов С. В. Контроль графов с отмеченными вершинами // Труды Донецк гос. тех. ун-та, сер.: Выч. Техника и автоматика. — 2002. — Вып. 38. — С. 226–232.

Грунский И. С., Стёпкин А. В. Распознавание конечного графа коллективом агентов // Труды ИПММ НАН Украины. — 2009. — Т. 19. — С. 43–52.

Килибарда Г., Кудрявцев В. Б., Ушчумлич Щ. Независимые системы автоматов в лабиринтах // Дискретная математика. — 2003a. — Т. 15, Вып. 2. — С. 3–39.

Килибарда Г., Кудрявцев В. Б., Ушчумлич Щ. Коллективы автоматов в лабиринтах // Дискретная математика. — 2003b. — Т. 15, Вып. 3. — С. 3–40.

Кормен Т., Лейзерсон Ч. Алгоритмы: построение и анализ. — М.: МЦНМО, 2001. — 960 с.

Стёпкин А. В. Возможность и сложность распознавания графов тремя агентами // Таврический вестник информатики и математики. — 2012. — №1 (20). — С. 88–98.

Albers S., Henzinger M. R. Exploring unknown environments // SIAM Journal on Computing. — 2000. — 29(4). — P. 1164–1188.

Deng X., Papadimitriou C. H. Exploring an unknown graph // Journal of Graph Theory. — 1999. — 32(3). — P. 265–297.

Dudek G., Jenkin M., Milios E., Wilkes D. Map validation in a graphlike world // Proceedings of the 13th International Joint Conference on Artifical Intelligence (Chambery, France, August 1993). — San Fransisco: Morgan Kaufmann Publishers Inc., 1993. — P. 1648–1653.