УДК 519.5

©2012. **Е. А. Татаринов**

О ВЕРХНЕЙ ОЦЕНКЕ СЛОЖНОСТИ ВОССТАНОВЛЕНИЯ РЕЗУЛЬТИРУЮЩЕГО ГРАФА, ПОЛУЧЕННОГО СОЧЛЕНЕНИЕМ ГРАФОВ-КОМПОНЕНТ

Анализируются композиции графов из компонент. Композиции представлены правильными и неправильными сочленениями. Предлагается способ и формулы для подсчета верхней оценки сложности восстановления результирующего графа, по известным верхним оценкам сложности восстановления его компонент. Полученные формулы обобщают формулы, полученные ранее для частных видов графов квазиколец и квазициклов.

Ключевые слова: *восстановление графов, композиции графов, сложность восстановления графов, графы-компоненты.*

1. Введение. Анализ глобальных свойств графа одно из направлений, связанных с исследованием поведения автоматов в лабиринтах [1, 2]. Наиболее значимыми задачами из этой области являются задачи: самолокализации, проверки карты и полного восстановления графа [3]. Наибольший интерес представляет задача полного восстановления графа, поскольку решение этой задачи позволяет получить решение двух других задач. Поэтому задача полного восстановления графа является важной в теоретическом и прикладном плане. В данной работе рассматривается задача полного восстановления неизвестного графа при помощи блуждающего по нему агента [4]. Ранее был предложен метод восстановления графа при помощи построения на его вершинах нумерации явной или неявной [2], предложен Базовый Алгоритм (БА) [6], его модификации и эвристики [7, 8], в данной работе предполагается, что агент использует Модификации 1-4 (М1, М2, М3, М4) БА [7], а также модификации БА [8], которые будем здесь называть М5, М5.1, М5.2. Далее будем предполагать, что агент выполняет М5, М5.1, М5.2, когда подсчитывается количество используемых камней, и БА или его модификации – Модификации 1-4. Для Базового Алгоритма, его модификаций и эвристик были получены верхние оценки сложности восстановления графа. Все они выражаются через известные характеристики графа и являются достижимыми, однако, они достигаются на отдельных видах графов, множество которых не совпадает со всем классом исследуемых графов. Поэтому нахождение нового способа вычисления верхней оценки сложности восстанволения графа для заданного графа из исследуемого класса является важной и актуальной задачей.

В работе предложен новый способ подсчета верхней оценки сложности восстановления графа, этот способ заключается в том, что рассматривается композиция графов из компонент, для которых верхние оценки сложности уже известны. Эти композиции делятся на правильные и неправильные сочленения. Исследуются изменения сложности восстановления результирующего графа в зависимости от харак-

тера композиции. Даются формулы подсчета сложности для правильных и неправильных сочленений графов. Эти формулы являются обобщением соответствующих формул из [7, 9], полученных для частных видов графов квазидеревьев и квазициклов.

2. Необходимые определения и понятия. Пусть K – класс простых, неориентированных, связных графов, без петель и ератных ребер. Граф $G(V,E) \in K$, где V – множество вершин, мощности n, а $E \subseteq V \times V$ – множество ребер, мощности m. Тройку $((u,v),v)$ будем называть инцидентором ("точкой прикосновения") ребра (u,v) 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 и вместо нее наносится новая. 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 $\mu : L \to M$, где $M = \{w, b, r, 1..,p\}$, w интерпретируется как белый цвет (краска), r – красный, b – черный, а множество чисел $1,..,p$ интерпретируется как множество камней. Изоморфизмом графа G и графа H назовем такую биекцию $\phi : V_G \to V_H$, что $(u,v) \in E_G$ точно тогда, когда $(\phi(v), \phi(u)) \in E_H$. Изоморфные графы равны с точностью до обозначения вершин и раскраски их элементов.

Древесными ребрами называются ребра, порождающие дерево поиска при обходе графа в глубину, остальные ребра – обратными. Будем обозначать $T^*(n)$ верхнюю оценку временной сложности выполнения алгоритма. Будем говорить, что граф G принадлежит классу M_c^p, если он восстанавливается агентом с использованием p – камней и c – красок.

3. Постановка задачи. Пусть задан класс K графов конечных, неориентированных, без петель и кратных ребер, все элементы которых окрашены краской w.

Задан мобильный агент (A), который обладает следующими свойствами. Он передвигается по графу из вершины u в вершину v по ребру (v,u). При этом он может изменить окраску вершин u, v, ребра (v,u), инциденторов $((v,u),v),((v,u),u)$ метками из множества M. Находясь в вершине v 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 $O(v)$. A обладает конечной, неограниченно растущей внутренней памятью, в которой фиксируется результат его функционирования на каждом шаге, и, кроме того, выстраивается представление графа G, вначале неизвестного A, списками ребер и вершин.

Требуется разработать такой алгоритм функционирования A (т.е. передвижения по графу и раскраски его элементов), что он, будучи помещен в произвольную вершину произвольного неизвестного ему графа $G \in K$, через конечное число шагов построит граф H, изоморфный G, т.е. восстановит G.

4. Базовый Алгоритм и его модификации. Базовый алгоритм подробно описан в [6], напомним его. Предложенный 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, 11]. 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 красной краской при первом прохождении по ним, 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 черной краской. Обратные ребра – проходятся один раз.

Древесные ребра распознаются агентом при первом проходе агента по ним. При первом посещении вершины агент метит ее красной краской. 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 становится пустым, а все вершины черными.

Вход: граф $G \in K$ неизвестный агенту, все элементы G окрашены в w, агент помещен в произвольную вершину v.

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

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

1. *Сч*:= 1; $t := 1$; $k(t) :=$; $V_H := \{1\}$; $E_H := \emptyset$;

2. Агент красит: $\mu(v) := r$;

3. *if* $O(v)$ есть ребро (v, u), у которого $\mu(u, v) = w$ *then goto* 4 *else goto 5*

4. *if* $O(v)$ есть ребро (v, u), у которого $\mu(u, v) = w$ и $\mu(u) = \mu(v) = r$ *then ВОСТ*(v) *else ВПЕРЕД*(v)

5. *if* в $O(v)$ есть ребро (v, u), у которого $\mu((v, u), v) = r$ *then do НАЗАД*(v) *goto 3 end do*

6. Агент красит: $\mu(v) := b$;

ВПЕРЕД(v) – достраивание дерева и восстановление древесного ребра. При этом новая вершина добавляется в красный путь.

НАЗАД(v) – возврат по дереву, когда все ребра, инцидентные текущей вершине, восстановлены. При этом текущая вершина удаляется из красного пути.

ВОСТ(v) – восстановление обратного ребра. Базовый Алгоритм был модифицирован с целью уменьшения верхней оценки сложности восстановления графа, однако требовал использования дополнительных ресурсов или дополнительного анализа окрестности вершины, в которой находится агент. Подробно они описаны в [6-8]. Напомним в чем заключаются эти модификации.

Модификация 1 (М1). Агент разделяется на два агента: агент-исследователь (АИ) и агент-экспериментатор (АЭ).

АИ передвигается по графу, считывает и изменяет метки на элементах графа, передает сообщения АЭ и на основе разметки элементов графа выбирает дальнейшее направление движения.

АЭ по сообщениям от агента АИ строит представление исследуемого графа в виде списка смежности.

Модификация 2 (М2). Агент восстанавливает все обратные ребра, инцидентные одной вершине красного пути, за один проход по вершинам красного пути. Для этого ему необходим дополнительный камень.

Модификация 3 (М3). Обратные ребра вершины восстанавливаются после восстановления всех ее древесных ребер.

Модификация изменяет в БА только приоритет выполнения двух процедур *ВОСТ*(v), *ВПЕРЕД*(v).

Модификация 4 (М4). Множество обратных ребер разбивается на подмножества, ребра из которых восстанавливались за конечное число шагов.

Множество вершин разбивается на ординарные и неординарные. Неординарные вершины метятся камнями. При этом используется дополнительно не более чем p камней.

Модификация 5 (М5). Агент реализует нумерацию при помощи двух красок и набора попарно различных камней.

Эвристика 5.1 (М5.1). Уменьшает число используемых камней.

Выделяются два типа вершин для пометки, в которых не используются камни:

1. из которых не видно непомеченных;

2. из которых видно только одну непомеченную вершину;

Эвристика 5.2 (М5.2). Уменьшает число используемых камней.

Агент метит камнями все вершины, расположенные на расстоянии не более чем k от текущей вершины. При этом камни на расстоянии меньшем k – станут свободными.

5. Правильные и неправильные сочленения. Пусть заданы графы G_1 и $G_2 \in K$. Сочленнием графов G_1 и G_2 будем называть такое преобразование над G_1 и G_2, что в результате будет получен новый граф $G \in K$. Графы G_1 и G_2 будем называть компонентами результирующего графа G. Таким образом, сочленение

двух графов – это преобразование над графами, результирующий граф которого не выходит из рассматриваемого класса графов K. Преобразования, которые выведут результирующий графа из рассмариваемого класса графов K, не рассматриваются. Правильным сочленением $G = G_1 \odot G_2$ будем называть такое преобразование, что в результате получается граф G, в котором нет циклов, ребра которых принадлежат обоим компонентам G_1 и G_2. Если $G_1 = G_2$ то, будем рассматривать только те случаи, когда результирующий граф отличается от исходного. Т.е., в результате преобразования граф изменился. При этом сочленение будет правильным, если не образуются новые циклы в результирующем графе.

Неправильным сочленением $G = G_1 \otimes G_2$ будем называть такое преобразование, что в результате получается граф G, в котором есть хотя бы один цикл, ребра которого принадлежат обоим компонентам G_1 и G_2. Если $G_1 = G_2$, то будем рассматривать только те случаи, когда результирующий граф отличается от исходного. Т.е., в результате преобразования граф изменился. При этом сочленение будет неправильным, если образуются новые циклы в результирующем графе.

Далее будем предполагать, что для выполнения одного преобразования может быть добавлено константное количество ребер и/или вершин в результирующий граф. В противном случае будем рассматривать композицию преобразования исходного графа на самого себя и одного или нескольких других графов, в результате которого может быть добавлено более чем константное количество ребер и вершин.

Свойства правильных и неправильных сочленений сформулируем в виде лемм и теоремы.

Лемма 1. *Пусть $G_1, G_2 \in K$ и $G = G_1 \odot G_2$, тогда $T^*(G) \leq T^*(G_1) + T^*(G_2) + \alpha_0^*$, где α_0^* – количество шагов, которое необходимо для восстановления вершин и ребер, добавленных в результирующий граф, и которые не содержатся в его компонентах G_1 и G_2.*

Доказательство. В результате правильного сочленения в результирующем графе может быть часть вершин и ребер, принадлежащих компонентам G_1 и / или G_2, добавлены дополнительные ребра и вершины, по которым он может попасть только в одну из компонент G_1 или G_2. Если же при добавлении таких ребер возникли новые циклы, ребра которых содержатся в компоненте G_1 (G_2), то такие случаи будем рассматривать как $G = (H \otimes G_1) \odot G_2$ ($G = (H \otimes G_2) \odot G_1$), где H – граф, образованный ребрами, добавленными в результирующий граф G и не содержащимися в его компонентах G_1 и G_2.

Поскольку, сочленение правильное, то в графе не будет новых циклов при восстановлении обратных ребер и, возможно, будет удалена часть ребер из компонент. Следовательно, при восстановлении графа общее время восстановления будет ограничено суммарным временем восстановления каждой компоненты. Действительно, проход по древесным ребрам суммируется, поскольку они проходятся в прямом и обратном направлении и учтены при подсчете $T^*(G_1)$ и $T^*(G_2)$. Восстановление обратных ребер будет проходить в пределах одной компоненты, в силу правильности сочленения, и так же учитывается при подсчете $T^*(G_1)$ и $T^*(G_2)$. Время, необходимое для восстановления части G, которая не принадлежит ни G_1, ни G_2 учитывается

в α_0^*.

Таким образом, общее время восстановления графа G оценивается $T^*(G) \leq T^*(G_1) + T^*(G_2) + \alpha_0^*$. Что и требовалось показать. □

При этом, если $G = G_1 \odot G_1$, то $T^*(G) \leq T^*(G_1) + \alpha_0^*$.

Следствие 1. *Пусть $G_k \in K$,$k = 1,..,i$ и $G = G_1 \odot G_2 \odot .. \odot G_{i-1} \odot G_i$ тогда $T^*(G) \leq \sum_{k=1}^{i} T^*(G_k) + \sum_{k=1}^{i-1} \alpha_{k,0}^*$, где $\alpha_{k,0}^*$, $k = 1,..,i-1$ – количество шагов, которое необходимо для восстановления вершин и ребер, добавленных в результирующий граф, и которые не содержатся в его компонентах G_k и G_{k+1}.*

Заметим, что, если $G = G_1 \odot G_1 \odot .. \odot G_1 \odot G_1$, где в правильном сочленении участвуют i графов G_1, то $T^*(G) \leq T^*(G_1) + (i-1)\,\alpha_0^*$.

Лемма 2. *Пусть $G_1, G_2 \in K$ и $G = G_1 \otimes G_2$, тогда $T^*(G) \leq T^*(G_1) + T^*(G_2) + \alpha_0^* + \beta_0^*$, где α_0^* – количество шагов, которое необходимо для восстановления вершин и ребер, добавленных в результирующий граф, и которые не содержатся в его компонентах G_1 и G_2, а β_0^* – количество шагов, которое необходимо для восстановления новых циклов в G, ребра которых содержатся одновременно в обоих компонентах G_1 и G_2.*

Доказательство. При восстановлении результирующего графа G общее время его восстановления может быть ограничено суммарным временем восстановления его компонент. Так как не может быть затрачено меньше времени на восстановление результирующего графа, чем суммарное время восстановления его компонент. С образованием новых циклов, ребра которых принадлежат обоим компонентам, уже будут совсем другие циклы для восстановления обратных ребер. Для учета этих изменений необходимо количество шагов, которое будет учтено в β_0^*. Время, необходимое для восстановления части графа G, которая не содержится ни в G_1, ни в G_2, как и в лемме 1, учитывается в α_0^*. Если же эта часть образует новые циклы, ребра которых содержатся в компоненте G_1 (G_2), то такие случаи будем рассматривать как $G = (H \otimes G_1) \odot G_2$ ($G = (H \otimes G_2) \odot G_1$), где H – граф, образованный частью графа G, которая не содержится ни в G_1, ни в G_2.

Таким образом, общее время восстановления графа G оценивается $T^*(G) \leq T^*(G_1) + T^*(G_2) + \alpha_0^* + \beta_0^*$. Что и требовалось показать. □

При этом, если $G = G_1 \otimes G_1$, то $T^*(G) \leq T^*(G_1) + \alpha_0^* + \beta_0^*$.

Следствие 2. *Пусть $G_k \in K$, $k = 1,..,i$ и $G = G_1 \otimes G_2 \otimes .. \otimes G_{i-1} \otimes G_i$ тогда $T^*(G) \leq \sum_{k=1}^{i} T^*(G_k) + \sum_{k=1}^{i-1} \left(\alpha_{k,0}^* + \beta_{k,0}^*\right)$, где $\alpha_{k,0}^*$, $k = 1,..,i-1$ – количество шагов, которое необходимо для восстановления вершин и ребер, добавленных в результирующий граф, и которые не содержатся в его компонентах G_k и G_{k+1}, а $\beta_{k,0}^*$, $k = 1,..,i-1$ – количество шагов, которое необходимо для восстановления новых циклов в G, ребра которых содержатся одновременно в обоих компонентах G_k и G_{k+1}.*

Заметим, что, если $G = G_1 \otimes G_1 \otimes .. \otimes G_1 \otimes G_1$, где в неправильном сочленении участвуют i графов G_1, то $T^*(G) \leq T^*(G_1) + (i-1)\,(\alpha_0^* + \beta_0^*)$.

Лемма 3. *Пусть $G_1 \in M_c^{p_1}$, $G_2 \in M_c^{p_2}$ и $G = G_1 \odot G_2$, тогда $G \in M_c^{p}$, где $p \leq p_1 + p_2 + \alpha_1^*$, где α_1^* – количество камней, которое необходимо для восстановления*

вершин и ребер, которые были добавлены в результирующий граф G, и которые не содержатся в обоих компонентах G_1 и G_2.

Доказательство. Для восстановления части результирующего графа G, которая не содержится в его компонентах G_1 и G_2, потребуются дополнительные камни. Количество этих камней учтено в α_1^*. Если же эта часть графа образует новые циклы, ребра которых содержатся в компоненте G_1 (G_2), то такие случаи будем рассматривать как $G = (H \otimes G_1) \odot G_2$ ($G = (H \otimes G_2) \odot G_1$), где H – граф, образованный частью графа G, которая не содержится ни в G_1 ни в G_2.

Поскольку сочленение правильное, то не будет циклов содержащих ребра из обоих компонент G_1 и G_2. А значит, при пометке вершины камнем не появятся дополнительные ребра, которые не дадут освободить камень в тот момент времени, как если бы агент восстанавливал одну из компонент G_1 или G_2 в отдельности. Следовательно, при восстановлении каждой компоненты G_1 и G_2 графа G будет использовано не большее количество камней, чем необходимо для восстановления их в отдельности.

Таким образом, суммарное количество камней может быть оценено $p \le p_1 + p_2 + \alpha_1^*$. Что и требовалось показать. □

При этом, если $G = G_1 \odot G_1$, то $p \le p_1 + \alpha_1^*$.

Следствие 3. *Пусть $G_k \in M_c^{p_k}$, $k = 1, .., i$ и $G = G_1 \odot G_2 \odot .. \odot G_{i-1} \odot G_i$, тогда $G \in M_c^p$, где $p \le \sum_{k=1}^{i} p_k + \sum_{k=1}^{i-1} \alpha_{k,1}^*$, и $\alpha_{k,1}^*$, $k = 1, .., i-1$ – количество камней, которое необходимо для восстановления вершин и ребер, которые были добавлены в результирующий граф G, и которые не содержатся в обоих компонентах G_k и G_{k+1}.*

Заметим, что, если $G = G_1 \odot G_1 \odot .. \odot G_1 \odot G_1$, $G_1 \in M_c^{p_1}$, где в правильном сочленении участвуют i графов G_1, то $G \in M_c^p$, где $p \le p_1 + (i-1)\,\alpha_1^*$.

Лемма 4. *Пусть $G_1 \in M_c^{p_1}$, $G_2 \in M_c^{p_2}$ и $G = G_1 \otimes G_2$, тогда $G \in M_c^p$, где $p \le p_1 + p_2 + \alpha_1^* + \beta_1^*$, где α_1^* – количество камней, которое необходимо для восстановления вершин и ребер, которые были добавлены в результирующий граф G, и которые не содержатся в обоих компонентах G_1 и G_2, а β_1^* – количество камней, которое необходимо для восстановления новых циклов в G, ребра которых содержатся одновременно в обоих компонентах G_1 и G_2.*

Доказательство. Для восстановления части результирующего графа G, которая не содержится в его компонентах G_1 и G_2, потребуются дополнительные камни. Количество этих камней учтено в α_1^*. Если же эта часть образует новые циклы, ребра которых содержатся в компоненте G_1 (G_2), то такие случаи будем рассматривать как $G = (H \otimes G_1) \odot G_2$ ($G = (H \otimes G_2) \odot G_1$), где H – граф, образованный частью графа G, которая не содержится ни в G_1, ни в G_2.

Поскольку сочленение неправильное, то будут циклы содержащие ребра из обоих компонент G_1 и G_2. А значит, при пометке вершины камнем появятся дополнительные ребра, которые не дадут освободить камень, в тот момент времени, как если бы агент восстанавливал одну из компонент G_1 или G_2 в отдельности, а следовательно при восстановлении каждой компоненты G_1 и G_2 графа G будет использовано большее количество камней, чем необходимо для восстановления их в отдельности.

Учитывая в β_1^* дополнительные камни, необходимые для восстановления каждой компоненты в связи с появившимися дополнительными циклами, суммарное количество камей может быть оценено $p \leq p_1 + p_2 + \alpha_1^* + \beta_1^*$. Что и требовалось показать. □

При этом, если $G = G_1 \otimes G_1$, $G_1 \in M_{c_1}^{p_1}$, то $p \leq p_1 + \alpha_1^* + \beta_1^*$

Следствие 4. *Пусть* $G_k \in M_{c_k}^{p_k}$, $k = 1, .., i$ *и* $G = G_1 \otimes G_2 \otimes .. \otimes G_{i-1} \otimes G_i$, *тогда* $G \in M_c^p$, *где* $p \leq \sum_{k=1}^{i} p_k + \sum_{k=1}^{i-1} \beta_{k,1}^*$, *где* $\beta_{k,1}^*$, $k = 1, .., i-1$ – *количество камней, которое необходимо для восстановления новых циклов в* G, *ребра которых содержатся одновременно в обоих компонентах* G_k *и* G_{k+1}.

Заметим, что, если $G = G_1 \otimes G_1 .. G_1 \otimes G_1$, $G_1 \in M_c^{p_1}$, где в правильном сочленении участвуют i графов G_1, то $G \in M_c^p$, где $p \leq p_1 + (i-1)(\alpha_1^* + \beta_1^*)$.

Лемма 5. *Если в результате удаления ребра из графа* $G \in K$, *результирующий граф не выходит из класса* K, *то он может быть представлен в виде правильного сочленение исходного графа* G *на самого себя.*

Доказательство. Действительно, при удалении ребра в результирующем графе не могут образоваться новые циклы, следовательно, это правильное сочленение. Таким образом, удаление ребра может быть представлено в виде $G \odot G$. Что и требовалось показать. □

Лемма 6. *Если в результате добавления ребра в граф* $G \in K$, *результирующий граф не выходит из класса* K, *то он может быть представлен в виде правильного или неправильного сочленения исходного графа* G *на самого себя, или правильного или неправильного сочленения* $G_1, G_2 \in K$.

Доказательство. При добавлении ребра в граф G могут, как образоваться новые циклы, так и не образоваться. В первом случае добавление ребра может быть представлено в виде $G \odot G$, а во-втором – в виде $G \otimes G$.

При добавлении ребра в граф $G = G_1 \odot G_2$ могут, как образоваться новые циклы, ребра которых содержатся в обоих компонентах G_1 и G_2, так и не образоваться. В первом случае $G = G \odot G$, а во-втором – в виде $G = G \otimes G$.

При добавлении ребра в граф $G = G_1 \otimes G_2$ могут, как образоваться новые циклы, ребра которых содержатся в обоих компонентах G_1 и G_2, так и не образоваться. В первом случае $G = G \odot G$, а во-втором – в виде $G = G \otimes G$.

Что и требовалось показать.□

Теорема 1. *Любая операция* Δ, *которая не выводит результирующий граф из класса* K, *может быть представлена в виде последовательного применения правильных и неправильных сочленений к множеству графов* $G_1, G_2, .., G_i \in K$.

Доказательство. Пусть Δ некоторая операция, которая не выводит результирующий граф из класса K. Результирующий граф G будет состоять из всех или части вершин его компонент (в случае если Δ унарная операция, то компонентами результирующего графа будет исходный граф). То есть, при выполнении операции результирующий граф содержит в себе все или часть вершин и / или ребер его компонент (часть из них может быть удалена, а часть новых – добавлена). Поскольку рассматриваются связные графы, то по сути своей удаление и добавление

новой вершины неразрывно связано с добавлением или удалением ребра, которым она соединяется с другой вершиной компонент графа. А в случае добавления новой вершины, не принадлежащей компонентам графа, она все равно должна быть соединена ребром с уже существующей в графе вершиной. Таким образом, любая операция Δ, которая не выводит результирующий граф из класса K, может быть представлена последовательным добавлением или удалением ребер. По леммам 5 и 6, Δ может, быть представлена в виде последовательного применения правильных и неправильных сочленений к множеству графов $G_1, G_2, .., G_i$. $\square$

Следствие 5. *Пусть* $G \in K$, *тогда* $T^*(G) \leq \sum_{k=1}^{i} T^*(G'_k) + \sum_{k=1}^{i-1} \left(\alpha^*_{k,0} + \beta^*_{k,0}\right)$, *где* $G'_1, G'_2, .., G'_i \in K$ *и* $G \in M_c^p$, *где* $G'_k \in M_c^{p_k}$, $k = 1, .., i$ *и* $p \leq \sum_{k=1}^{i} p_k + \sum_{k=1}^{i-1} \left(\alpha^*_{k,1} + \beta^*_{k,1}\right)$, *при этом, если компоненты* G_k *и* G_{k+1} *дают правильное сочленение и* $\beta^*_{k,l} = 0$.

Доказательство. Поскольку любой граф $G \in K$ может быть представлен в виде последовательного выполнения некоторых операций Δ_k, $k = 1, .., l$, над множеством графов $G_1, G_2, .., G_{l+1}$, то каждая пара $G_k \Delta G_{k+1}$ может быть представлена в виде последовательного применения правильных и неправильных сочленений к некоторому множеству графов. Следовательно, любой граф $G \in K$ может быть представлен в виде последовательного применения правильных и неправильных сочленений к множеству графов $G'_1, G'_2, .., G'_i \in K$, по теореме 1. $\square$

Ранее было показано [7, 9], что для квазидеревьев (класс T^k) и квазиколец (класс S^k) сложность восстановления выражается через сложность восстановления их компонент. В общем виде это можно сформулировать в виде следующих теорем.

Теорема 2. *При восстановлении графа* $G \in T^k \cup S^k$ *агентом верхняя оценка временной сложности равна* $T^*(n) = \sum_{j=1}^{k} T^*(n_j) + \alpha_0 + \beta_0$, *где* α_0 *и* β_0 *могут быть найдены из приведенной ниже таблицы, а* $T^*(n_j)$ – *сложность восстановления* $G_j \in K$ – j*-ой компоненты графа* G. *При этом результирующий граф может быть получен путем выполнения операции сочленения или соединения мостом.*

$G \in T^k$	Сочленение		Соединение мостом	
	α_0	β_0	α_0	β_0
БА,М2	0	0	$2\sum_{i=1}^{k-1} l_i$	0
$G \in S^k$	Сочленение		Соединение мостом	
	α_0	β_0	α_0	β_0
БА	0	$O(\lvert E_1 \rvert n)$	$2\sum_{i=1}^{k-1} l_i$	$O(\lvert E_1 \rvert n)$
М2	0	$O(\lvert V_1 \rvert n)$	$2\sum_{i=1}^{k-1} l_i$	$O(\lvert V_1 \rvert n)$

В таблице l_i – длина i-ого моста, E_1 и V_1 множество вершин компоненты графа, из которой агент начал восстановление результирующего графа G, принадлежащего классу S^k

Теорема 3. *При восстановлении графа $G \in T^k \cup S^k$ верхняя оценка числа камней равна $\sum_{j=1}^{k} p_j + \alpha_1 + \beta_1$, где α_1 и β_1 могут быть найдены из приведенной ниже таблицы, а p_j – количество камней, необходимых для восстановления $G_j \in K$ – j-ой компоненты графа G. При этом результирующий граф может быть получен путем выполнения операции сочленения или соединения мостом.*

$G \in T^k$	Сочленение		Соединение мостом	
	α_1	β_1	α_1	β_1
M5,M5.2	0	0	$(k-2)+4$	
M5.1	0	0	$(k-2)+2$	
$G \in S^k$	Сочленение		Соединение мостом	
	α_1	β_1	α_1	β_1
M5,M5.2	0	p_1	$(k-2)+4$	p_1
M5.1	0	p_1	$(k-2)+2$	p_1

В таблице p_1 – количество камней, необходимых агенту для восстановления компоненты графа, из которой агент начал восстановление результирующего графа G, принадлежащего классу S^k.

Таким образом, $G \in T^k$ является частным случаем правильного сочленения, а $G \in S^k$ – неправильного.

6. Выводы. В работе предложены правильные и неправильные сочленения графов-компонент. Подсчитывается верхняя оценка сложности восстановления результирующего графа, по известным верхним оценкам сложности восстановления компонент, при достаточно широких предположениях о свойствах сочленения.

1. *Килибарда Г., Кудрявцев В.Б., Ушчумлич Щ.* Коллективы автоматов в лабиринтах // Дискретная математика. – 2003. – Т. 15. – Вып. 3. – С. 3-40.
2. *Килибарда Г., Кудрявцев В.Б., Ушчумлич Щ.* Независимые системы автоматов в лабиринтах // Дискретная математика. – 2003. – Т. 15. – Вып. 2. – С. 3-39.
3. *Dudek G., Jenkin M., Milios E., Wilkes D.* Map validation and Robot Self-Location in a Graph-Like World // Robotics and Autonomous Systems. – 1997. – Vol. 22.
4. *Dudek G., Jenkin M.* Computational principles of mobile robotic // Cambridge Univ. Press. – 2000. – 280 p.
5. *Татаринов Е.А.* М-нумерация, как метод распознавания графов // Збірник наукових праць "Питання прикладної математики і математичного моделювання". – Дніпропетровськ: Вид. Дніпр. нац. ун-та. – 2010. – С. 260-272.
6. *Грунский И.С., Татаринов Е.А.* Распознавание конечного графа блуждающим по нему агентом // Вестник Донецкого университета. Серия А. Естественные науки. – 2009. – Вып. 1. – С. 492-497.
7. *Татаринов Е.А.* Базовый алгоритм восстановления графа // Труды ИПММ НАН Украины. – 2010. – Т. 21. – С. 492-497.
8. *Татаринов Е.А.* Восстановление графа при помощи камней //Збірник наукових праць "Питання прикладної математики і математичного моделювання". – Дніпропетровськ: Вид. Дніпр. нац. ун-та. – 2011. – С. 232-255.

9. *Татаринов Е.А.* Сложность восстановления графов, являющихся квазикольцами и квазидеревьями // Труды ИПММ НАН Украины. – 2011. – Т. 23. – С. 202-212.
10. *Кормен Т., Лейзерсон Ч., Ривест Р.* Алгоритмы: построение и анализ. – М.: МЦНМО, 2001. – 960 с.
11. *Касьянов В.Н., Евстигнеев В.А.* Графы в программировании: обработка, визуализация и применение. – СПБ.: БХВ – Петербург, 2003. – 1104 с.

E. A. Tatarinov

About upper bound reconstruction resulting graph obtained graphs-component connections.

The components compositions of graphs are analyzed. Compositions can be regular and not regular connections. Provides the method and formulas to calculate the upper bound of the resulting graph reconstruction,by the known upper bounds of its components reconstruction. The formulas generalize the formulas obtained earlier for particular types of graphs.

Keywords: *graph reconstruction, composition graphs, reconstruction graphs complexity, graph-component.*

Є. О. Татаринов

Про верхню оцінку складності відновлення результуючого графа, отриманого зчленуванням графів-компонент.

Аналізуються композиції графів із компонент. Композиції представлено правильними та неправильними зчленуваннями. Пропонується спосіб і формули для підрахунку верхньої оцінки складності відновлення результуючого графа, за відомими верхніми оцінками складності відновлення його компонент. Здобуті формули узагальнюють формули, отримані раніше для окремих видів графів квазікілець і квазіциклів.

Ключові слова: *відновлення графів, композиції графів, складність відновлення графів, графи-компоненти.*

Ин-т прикл. математики и механики НАН Украины, Донецк
MDgerelo@yandex.ru

Получено 02.12.11