Об обходе графов автоматами с одной нестираемой краской

Д.В. Голубев

В работе рассматривается задача о построении для заданного натурального k автомата с «нестираемой краской» (печатающего один нестираемый символ в вершинах графа), обходящего произвольные графы с ограничением на степень вершины $deg(v) \leqslant k$. Кроме того, построен автоматный (реализуемый универсальным автоматом с нестираемой краской) способ раскраски лабиринта, такой что другой универсальный автомат, но уже без краски может обойти любой из лабиринтов, покрашенных этим способом.

1. Определения r- и c-лабиринтов, иерархия классов лабиринтов, усиление автомата красками

В работе [2] дано определение r- и c-лабиринтов через понятие графоида (*graphoid*), в нашем случае удобнее пользоваться понятием графа.

В дальнейшем будут рассматриваться только связные простые (без параллельных ребер и петель) ориентированные графы, заданные множеством вершин V и E – набором упорядоченных пар вершин (v_i, v_j), причем если $(v_i, v_j) \in E$, то и $(v_j, v_i) \in E$. Кроме того, будем считать, что $|V| > 1$.

Пусть $G(V, E)$ – граф, V – множество вершин, E – набор ребер, тогда обозначим $E(v) = \{(v_i, v_j) | v_i = v\}$ – множество ребер, выходя-

щих из v.

Пусть M – непустое конечное множество. Скажем, что r – вращение на M, если r – биекция M на себя, такая что $r^{|M|}(m) = m$ для любого $m \in M$ и при этом $r^i(m) \neq r^j(m)$, $1 \leqslant i \neq j < |M|$. То есть r можно задать последовательностью $(m, r(m), r^2(m), \ldots, r^{|M|-1}(m))$.

Определение. Пусть $G(V, E)$, где $V = v_1 \ldots v_n$ – граф, и в каждой вершине $v \in V$ задано вращение r_v на множестве $E(v)$. Назовем $R = (r_{v_1}, r_{v_2}, \ldots, r_{v_n})$ системой вращений графа G, а пару (G, R) – r-графом. Обозначим $R^y(v_i, v_j) = r^y_{v_i}(v_i, v_j)$.

Определение. Пусть задан граф $G(V, E)$ и D – некоторое конечное непустое множество. Будем называть системой направлений графа G (с множеством направлений D) функцию $c : E \to D$ такую, что при $e_1 \neq e_2$, $e_1 \in E(v), e_2 \in E(v)$ выполнено $c(e_1) \neq c(e_2)$. Отсюда, в частности, следует, что $|D| \geqslant deg(v)$ для любой вершины v. Обозначим $D(v) = \{d \in D | d = c(e), \text{ при } e \in E(v)\}$. Тогда тройку (G, D, c) будем называть c-графом.

Очевидно, зафиксировав произвольное вращение в D, мы автоматически получим вращение на каждом $D(v)$, и тем самым систему вращений на $E(v)$, то есть получим r-граф. В этом смысле c-графы являются частным случаем r-графов.

Теперь опишем, как связаны автоматы и графы в понятии об автоматах в лабиринтах. Основная идея такова – в каждый момент времени автомат находится в некоторой вершине и способен пользоваться некоторой локальной относительно этой вершины информацией для выбора следующей вершины.

Для того, чтобы перемещаться в лабиринтах, автомат должен иметь специальную структуру входа и выхода, которая должна быть согласована с типом лабиринта. Кроме того, автомат может использовать некоторые специальные средства для обхода лабиринта (например, метки, камни, коллективный обход и т.п.), что также должно отражаться на структуре входа и выхода. Определим простейший случай конечного инициального автомата, способного перемещаться по лабиринту в обоих из рассмотренных случаев графов.

Рассмотрим конечный инициальный автомат $A_{q_0} = (X, Y, Q, \varphi, \psi, q_0)$.

Здесь X – конечное входное множество, Y – конечное выходное множество, Q – конечное множество состояний, φ – функция переходов $\varphi : X \times Q \to Q$, ψ – функция выхода $\psi : X \times Q \to Y$ и $q_0 \in Q$ – начальное состояние.

Для r- и c-лабиринтов автоматы будут различаться структурой входа и выхода.

В случае c-лабиринтов $X = \{(S, d) : S \subseteq D, d \in S\}$, $Y = D \cup \{-\}$, $\psi((S, d), q) \in S \cup \{-\}$ для всех $((S, d), q) \in X \times Q$.

Автомат связан с лабиринтом следующим образом. В начальный момент $t = 0$ он связан с некоторой вершиной графа v_0 и ему на вход подается множество направлений $S(v_0)$, выходящих из этой вершины, и произвольное направление $d_0 \in S(v_0)$.

Выход автомата интерпретируется следующим образом:

1) $\psi((S(v_t), d_t), q_t) = d \in S(v_t)$, тогда автомат перемещается в v_{t+1} по ребру с направлением d и в следующий момент на вход автомата подается $S(v_{t+1})$ и направление ребра $d_{t+1} = c((v_{t+1}, v_t))$.

2) $\psi((S(v_t), d_t), q_t) = \{-\}$, тогда автомат остается в той же вершине v_t и на вход подается та же пара $S(v_{t+1}, d_{t+1})$, где $v_{t+1} = v_t$ и $d_{t+1} = d_t$.

Более точно, назовем конфигурацией автомата на c-графе $(G(V, E), D, c)$ тройку (v, d, q), где $q \in Q$, $v \in V$, $d \in D(v)$. Таким образом, в каждый момент времени происходит изменение конфигурации. Для c-лабиринтов, как видно из определения функции автомата, существенна информация не только о возможных направлениях дальнейшего движения автомата, но и о том, по какому ребру автомат попал в данную вершину.

В случае r-лабиринтов будем рассматривать графы с $deg(v) \leqslant k$. Тогда входное и выходное множество автомата будут: $X = \{1, 2 \ldots k\}$ и $Y = \{0, 1, \ldots, k-1, -\}$, то есть автомат, находясь в некоторой вершине r-графа G, видит только, сколько ребер выходит из данной вершины.

В начальный момент $t = 0$ автомат связан с некоторой вершиной v_0 и некоторым ребром $e_0 \in E(v_0)$ графа G. В момент t автомат

связан с вершиной v_t и ребром e_t, которое определяется согласно указанному ниже правилу.

Выход автомата $y \in \{0, 1, \ldots, k-1\}$ интерпретируется как выбор ребра $(v_t, v_{t+1}) = R^y(e_t)$ и переход по нему в следующую вершину v_{t+1} с новым ребром $e_{t+1} = (v_{t+1}, v_t)$. Если на выход подан $\{-\}$, то автомат остается в $v_{t+1} = v_t$ и $e_{t+1} = e_t$.

Более точно, назовем конфигурацией автомата A на r-графе $G(V, E)$ тройку (v, e, q), где $q \in Q$, $v \in V$, $e \in E(v)$. Конфигурация определяет положение автомата, его текущеее состояние и ребро, необходимое для интерпретации выхода. В каждый момент времени происходит изменение конфигурации по указанному закону, и фактически автомат может выбирать лишь направление относительно ребра, по которому он пришел.

Система вращений в такой интерпретации локально лишает ребра, выходящие из одной вершины, уникальности. На первый план выходит их взаимное расположение, а не идентификация каждого ребра. Это обстоятельство в ряде случаев заставляет сужать класс лабиринтов, вводя доступную автомату дополнительную информацию, например, метки на ребрах, подчиненные определенным правилам.

Кроме того, в связи с отрицательными результатами, возникшими при решении многих задач, посвященных автоматам в лабиринтах, возникла потребность в более информативных подклассах r-лабиринтов, чем просто выделение c-лабиринтов.

Традиционным и хорошо изученным классом лабиринтов являются плоские лабиринты. Особое место среди них занимают прямоугольные плоские лабиринты с системой направлений $north, east, south, west$, причем такие, что противоположные ребра должны иметь противоположные метки ($north, south$, например). Кроме того, должно допускаться плоское геометрическое представление графа в виде множества вершин на целочисленной решетке, соединенных непересекающимися интервалами (соответствующими неориентированным ребрам), параллельными координатным прямым. Причём в этом случае система направлений на ориентированных ребрах согласуется с направлениями координатных осей (то есть, единые направления для north, south, west, east относительно координатных направлений). Аналогично можно ввести прямоугольные лабиринты в $\mathbb{R}^3, \ldots, \mathbb{R}^n$ с вершинами в целочисленной решётке $\mathbb{Z}^3, \ldots, \mathbb{Z}^n$. Несложно видеть, что все эти лабиринты являются c-лабиринтами с дополнительными требованиями на метки ребер.

Основной задачей об автоматах в лабиринтах является задача о построении конечного автомата, обходящего заданный класс лабиринтов, то есть автомата, который в начальном состоянии будучи установленным в любую вершину любого лабиринта из этого класса, с любым начальным ребром (если это требуется), должен к некоторому моменту T посетить все вершины соответствующего лабиринта.

Уже на плоских лабиринтах при попытке обойти их автоматом без вспомогательных средств возникают трудности, и согласно теореме Будаха [1] для любого автомата найдется плоский прямоугольный шахматный лабиринт, который данный автомат не обходит.

Таким образом, вопрос об обходе более широких классов лабиринтов решается отрицательно, в том числе для r- и c-лабиринтов с ограничением на степень вершины графа.

В связи с этим большой интерес представляет вопрос, а нельзя ли, добавив некоторые дополнительные возможности автомату (желательно минимальные), добиться разрешимости задачи обхода в как можно более широких классах лабиринтов.

Одним из наиболее естественных подходов является придание автомату возможности расставлять метки на вершинах или ребрах лабиринта. Здесь мы рассмотрим случай меток на вершинах, то есть в каждый момент времени каждой вершине графа будет приписан символ из $U = \{0, 1, \ldots, n\}$. Это означает, что в процессе работы автомата в лабиринте в каждый момент будет задана некоторая функция $u(v, t)$, определяющая цвет конкретной вершины. Обозначим $U(t)$ функцию, определяющую всю раскраску графа в данный момент времени.

2. Теорема об одной краске для трех задач об автоматах в лабиринтах

Здесь мы рассмотрим три близкие задачи об автоматах и лабиринтах:

1) Пусть вершинам графа G приписаны некоторые метки (цвета), и пусть автомат имеет возможность получать на входе цвет текущей вершины. Тогда существует ли автомат, обходящий произвольный лабиринт G из заданного класса хотя бы при одной из возможных вершинных раскрасок G (то есть данный автомат должен обходить любой лабиринт G из заданного класса при некоторой $U_G(0)$, причём $U_G(t) = U_G(0)$ для любого момента t)?

2) Пусть автомат имеет возможность (с помощью своего выхода) наносить цвета на вершины лабиринта и получать информацию о цвете вершины, где он находится. Тогда можно ли построить такой автомат, который обойдет произвольный, изначально окрашенный в один цвет ($u(v,0) = 0$), лабиринт из заданного класса?

Вообще говоря, из решения первой задачи не следует разрешимость второй, и наоборот.

3) Кроме того, если и первое и второе возможно, то существует ли автомат, способный раскрасить любой лабиринт из заданного класса и перейти в заключительное состояние так, что некоторый другой автомат сможет обойти любой такой раскрашенный лабиринт, имея возможность лишь считывать цвет текущей вершины.

Кроме вопроса о существовании таких автоматов возникает вопрос о минимальном числе цветов, необходимом для решения, а также, можно ли, если первоначальный цвет изменен, стирать краску или менять её на другую. Автомат, который может менять первоначальный цвет вершины $u(v) = 0$ на $1, \ldots, n$ мы назовем автоматом с n красками.

Пусть G_k^R – это все r-лабиринты с ограничением k на степень вершины, то есть любая вершина любого лабиринта из G_k^R имеет степень не более k.

Для определения автомата с красками необходимо добавить к

выходу автомата разряд, который будет интерпретироваться как нанесение нового цвета или сохранение существующего. В случае r-лабиринтов надо брать выходное множество Y' как прямое произведение Y и U – множества цветов (включая начальный цвет лабиринта). То есть на выходе будет пара $(y,u) \in Y \times U$, где второй элемент будет интерпретироваться как нанесение цвета на вершину, в которой данный автомат находится. Кроме того, будем считать, что автомат с красками может считывать цвет вершины, то есть в его входе должен присутствовать разряд, интерпретируемый как цвет, и входное множество X' также надо рассматривать как прямое произведение X и U, причем вход будет представлять из себя пару $(x,u) \in X \times U$. Соответственно функции φ и ψ будут определены как $\varphi : X' \times Q \to Q$, $\psi : X' \times Q \to Y'$. Кроме того, к конфигурации $(v(t), e(t), q(t))$ должна быть присоединена $U(t)$, то есть теперь конфигурация будет $(v(t), e(t), q(t), U(t))$.

В работе [3] построен автомат, решающий задачу 2) в случае плоских прямоугольных лабиринтов с помощью одной нестираемой краски (то есть $U = \{0,1\}$ и $\psi'((x,1),q) = (y,1)$ для любого $x \in X$, $q \in Q$). В данной работе показано, что для любого класса G_k^R, $k > 0$ все три вопроса имеют положительное разрешение с использованием одной нестираемой краски, и справедлива следующая теорема.

Теорема 1. *Для любого $k > 0$ существует конечный инициальный автомат $A^k(X', Y', Q_a, \varphi_a, \psi_a, q_a^0)$ с одной нестираемой краской, и $B^k(X', Y, Q_b, \varphi_b, \psi_b, q_b^0)$ – автомат, умеющий только считывать цвета вершин, такие, что:*

1) A^k обходит произвольный лабиринт G из G_k^R из произвольной начальной конфигурации $(v, e, q_a^0, U(0))$, где $u(v,0) = 0$.

2) После обхода лабиринта G автомат A^k переходит в заключительное состояние и останавливается в конфигурации $(v(T), e(T), q_{end}, U(T))$ в некоторый момент T.

3) B^k обходит лабиринт G при начальной конфигурации $(v(0), e(0), q_b^0, U'(0))$, где $v(0), e(0)$ – произвольные вершина и ребро, из неё выходящее, а $U'(0) = U(T)$.

Для доказательства будет построен алгоритм для автомата A^k. Кроме того, неявно при его построении будет использован алгоритм для автомата B^k, что будет прокомментировано после описания основного алгоритма.

3. Опорные вершины в графе

Определение. Расстоянием между вершинами графа v_1 и v_2 назовем длину минимального пути между v_1 и v_2 и обозначим ее $L(v_1, v_2)$. Причём $L(v, v) = 0$ по определению.

Определение. Назовем шаром $B_h(v)$ с центром в v и радиусом h множество вершин $B_h(v) = \{v_i | L(v_i, v) \leqslant h\}$.

Определение. $O_h(v) = B_h(v) \setminus B_{h-1}(v)$ – множество вершин, удаленных от v на h.

Будем считать, что вершинам графа приписаны метки из $U = \{0, 1\}$, причем вершины с меткой 0 будем считать непомеченными, а вершины с меткой 1 – помеченными. Пусть задан некоторый радиус шара $\varepsilon \geqslant 3$, тогда относительно этого ε введем определение опорной вершины.

Определение. Вершина v называется опорной типа $0 < m \leqslant \varepsilon - 2$ относительно ε, если:

1) Сама v помеченная.
2) Вершины из $O_1(v)$ не помечены.
3) Существует путь $(v_2, v_3, \ldots, v_{m+2})$, где $(v_i, v_{i+1}) \in E$, $v_i \in O_i(v)$, причем все v_i помечены.
4) Все вершины из $B_{m+3}(v)$, кроме $v, v_2, v_3, \ldots v_{m+2}$, не помечены.

Алгоритм будет основан на том, что автомат может, во-первых, распознать вершину как опорную, во-вторых, определить её тип, и в-третьих, пользуясь разметкой опорной вершины, установить локальный порядок на путях, ведущих от одной опорной вершины к другой. В результате, расширяя систему опорных вершин, автомат A^k обойдет весь лабиринт, причем так, что оставшаяся раскраска позволит обходить его автомату B^k уже не ставя дополнительных меток.

4. Алгоритм обхода графа с построением опорных вершин трех типов

Под обходом графа $G = (V, E)$ мы будем понимать последовательный выбор очередной вершины $v(t)$ согласно определенным правилам, исходя из начальной вершины $v(0)$, в результате которого все вершины G будут выбраны к некоторому шагу T.

Здесь будет приведен алгоритм обхода графа $G(V, E)$ из начальной вершины v_0 посредством перехода от одной вершины v_t к следующей v_{t+1} по ребру (v_t, v_{t+1}) с помощью пометки некоторых вершин в процессе работы алгоритма одной нестираемой краской. То есть вершинам графа будут приписаны два цвета; в начальный момент это один цвет для всех вершин, который можно изменить. Про вершины с таким цветом мы будем говорить, что они не помечены. Второй цвет будет присваиваться вершинам в процессе работы алгоритма, и этот цвет уже не может быть изменен. Про вершины со вторым цветом мы будем соответственно говорить, что они помечены.

Далее, убедившись в том, что данный алгоритм позволяет обойти граф, покажем, что он реализуется автоматом с одной нестираемой краской в случае, когда G снабжен системой вращений r-лабиринта.

Введем следующие обозначения.

$\varepsilon \geqslant 12, \varepsilon \in \mathbb{N}$ – основной параметр алгоритма обхода.

Пусть $v \in V, S \subseteq V$, тогда определим $L(S, v)$ расстояние от вершины до множества следующим образом.

1. При $v \in S$, $L(S, v) = 0$.
2. При $S = \emptyset$, $L(S, v) = \varepsilon + 1$.
3. $L(S, v) = \min\limits_{v' \in S} L(v', v)$.

Алгоритм разметки опорных вершин состоит в следующем.

Шаг 1. а) Обойдем $B_\varepsilon(v_0)$. б) Если возможно, пометим v_0 как вершину типа 1 и рассмотрим множество $S^1 = \{v_0\}$. Если невозможно, то алгоритм завершает работу (граф обойден).

Шаг 2. Если $S^i = \{v_1^i, \ldots, v_{n(i)}^i\}$ сформировано, то формируем S^{i+1} следующим образом:

а) S^{i+1} изначально пусто.

б) Рассмотрим $O_\varepsilon(v_1^i)$ и проверим существование $v^{i+1} \in O_\varepsilon(v_1^i)$, такого что:

α) $L(S^i, v^{i+1}) = \varepsilon$.

β) $L(S^{i+1}, v^{i+1}) \geqslant \varepsilon$.

γ) При $i > 1$ $L(S^{i-1}, v^{i+1}) > \varepsilon$.

δ) Для любой опорной вершины v произвольного типа $L(v, v^{i+1}) \geqslant \varepsilon$.

Если такое v^{i+1} существует, то добавляем его в S^{i+1}, метим v^{i+1} как опорную вершину типа $(i+1) \bmod 3 + 3((i+2) \bmod 3)(i \bmod 3)/2$ (это возможно из условия δ), после чего обходим $B_\varepsilon(v^{i+1})$.

в) повторяем шаг б), пока можно увеличить S^{i+1}.

г) переходим к следующему элементу из S^i и повторяем для него б) и в).

д) повторяем г) пока не переберем все элементы S^i.

Шаг 3. Если S^{i+1} не пусто, то формируем S^{i+2}, возвращаясь к шагу 2). Если же очередное S^{i+1} пусто, то алгоритм завершает свою работу.

Теорема 2. *Алгоритм разметки опорных вершин обходит граф G.*

Доказательство. Если $O_3(v_0) = \emptyset$, то утверждение очевидно.

Пусть $O_3(v_0) \neq \emptyset$, тогда существует возможность разметить $B_\varepsilon(v_0)$ так, что будет вершиной типа 1. Для этого можно рассмотреть любой путь длины три (v_0, v_1, v_2, v_3) из v_0 в $v_3 \in O_3(v_0)$ и пометить соответственно v_0, v_2, v_3. То есть в этом случае будет построено как минимум S^1.

Так как граф конечный и $S^i \cap S^j = \emptyset$ для $i \neq j$, то были последовательно построены $S^1, S^2, \ldots, S^n$ – непустые множества (где n может равняться 1), после чего S^{n+1} осталось пустым и алгоритм завершил работу. Предположим, что существует вершина v'', которая не была обойдена ни на одном шаге 2б) в результате обхода $B_\varepsilon(v_t)$ некоторой опорной вершины v_t. Так как граф связный, то существует путь из v_0 в v''. Рассмотрим вершину v', обладающую тем же свойством, что и v'' и ближайшую к v_0, а также вершину v_ε, соседнюю с v', которая была обойдена на шаге 2б) при обходе некоторого шара $B_\varepsilon(v_t)$. v_ε не может принадлежать ни одному из $S^1, \ldots, S^n$, иначе v' была бы обойдена. Более того $L(S^j, v_\varepsilon) \geqslant \varepsilon$ для любого j, иначе v' была бы обойдена. С другой стороны, из того, что v' обойдена, следует, что существует i, что $L(S^i, v') = \varepsilon$. То есть при этом i для вершины v' были выполнены условия α),β),δ) пункта 2б), но она не была выбрана в качестве опорной, то есть не было выполнено γ) и $i > 1$. Из $L(S^j, v_\varepsilon) \geqslant \varepsilon$ имеет место равенство $L(S^{i-1}, v_\varepsilon) = \varepsilon$, то есть существует опорная вершина где равенство имеет место. Аналогично, рассмотрев S^{i-1} имеем $L(S^{i-2}, v_\varepsilon) = \varepsilon$ и т.д. до $L(S^1, v_\varepsilon) = \varepsilon$ при условии $L(S^2, v_\varepsilon) \geqslant \varepsilon$, то есть v_ε должна была быть выбранной в S^2 как удовлетворяющая условиям 2б), и ее окрестность должна быть обойдена, то есть мы пришли к противоречию. Теорема доказана.

Утверждение 1. *Пусть $S^1, \ldots, S^n$ – множества, полученные на каком-то шаге 3) алгоритма.*

1) Если $m, i \in \{1, \ldots, n\}$ и $m \neq i, m \neq i-1, m \neq i+1, v \in S^m$, то $L(S^i, v) > \varepsilon$.

2) Если $m, i \in \{1, \ldots, n\}$ и $m = i \pm 1, v \in S^m$, то существуют $v_i \in S^i$, что $L(v, v_i) = \varepsilon$.

Доказательство. 2). Доказательство второй части очевидно из построения.

1) Предположим, что существует $m > i+1$, что $L(v_i, v_m) \leqslant \varepsilon$ для $v_m \in S^m, v_i \in S^i$. Выберем среди таких i минимальное. Так как v_m было построено на некотором шаге и прошло проверку δ) относительно v_i, то верны A) $L(v_m, v_i) = \varepsilon$ и B) $L(S^j, v^i) \geqslant \varepsilon$ для любого $j < m$.

Рассмотрим шаг 2б) для вершины v_i при построении S^{i+1}. Так как v_m не попал в S^{i+1}, то следовательно, не было выполнено хотя бы одно из условий α),β),γ),δ) в момент, когда рассматривалось v_i и $S_0^{i+1} \subset S^{i+1}$.

Проверим эти условия.

α) выполнено, так как $L(v_m, v_i) = \varepsilon$ из A);

β) выполнено из B) при $j = i+1$;

γ) При $i < 2$ не влияет на формирование S^{i+1}. При $i \geqslant 2$ $L(S^{i-1}, v_m) > \varepsilon$ из В) и того, что i минимально, то есть условие выполнено;

δ) выполнено из В).

То есть все условия были выполнены и v_m должно было войти в S^{i+1}, следовательно получено противоречие, так как $L(S^{i+1}, v_m) \geqslant \varepsilon$.

Аналогично проверяется случай $m < i - 1$.

Утверждение доказано полностью.

Для построения автомата, реализующего этот алгоритм, достаточно построить автоматы для выполнения следующих шагов:

0) Автомат, обходит $B_\varepsilon(v)$, возвращается в v, при $O_3(v) \neq \emptyset$ и переходит в заключительное состояние q^y, при $O_3(v) = \emptyset$ переходит в заключительное состояние q^n.

1) Автомат из вершины v и начального состояния q^0 обходит $B_\varepsilon(v)$ и возвращается в v в состояние q^s.

2) Автомат, находясь в вершине v и начальном состоянии q^0, размечает $B_\varepsilon(v)$ согласно заданному типу v и возвращается в v в состояние q^s. Будем считать, что шаг применяется, только если $B_\varepsilon(v)$ можно разметить. Из построенния алгоритма существование возможности разметки будет очевидно.

3) По уже нанесенной разметке на вершины лабиринта G автомат, находясь в вершине v, зная тип S^{i+1}, проверяет условия пункта 2б) и возвращается в нее, характеризуя проверку одним из двух заключительных состояний q^y,q^n, если вершина прошла и не прошла проверку соответственно.

4) Автомат, зная тип S^{i+1}, находясь в начальной вершине v и начальном состоянии q^0, выполняет упорядоченный обход всех кандидатов на присоединение к S^{i+1} и останавливается после этого обхода в вершине v_0, причем будем считать, что он имеет сигнальные состояния q^{kl} означающие, что он находится в очередном кандидате на присоединение. В конце работы он дожен оказаться в v_0 в заключительном состоянии q^s

Более того, потребуем от первых четырех автоматов, чтобы они не только возвращались в исходную вершину, но и сохраняли в полученной конфигурации и начальное ребро.

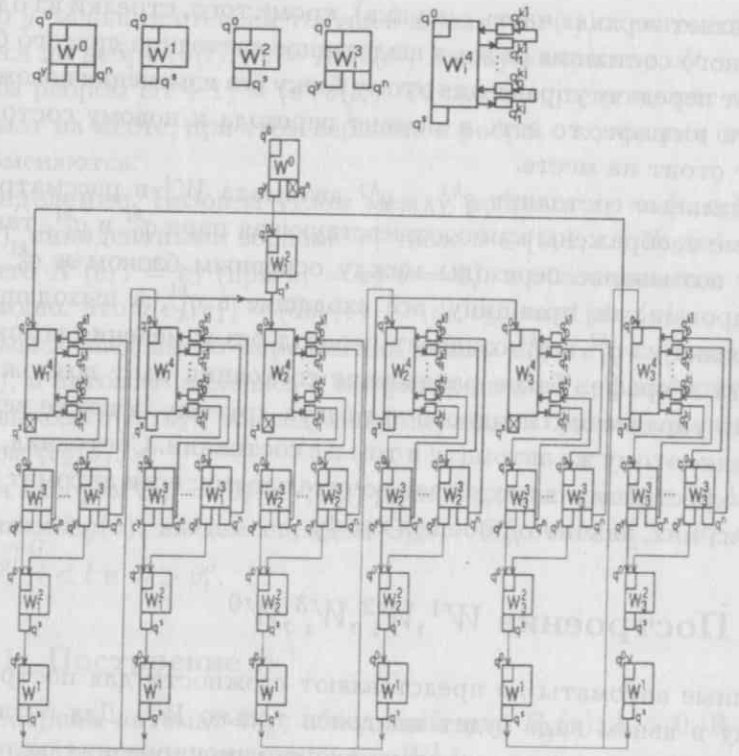

Рис. 1. Автомат исполняющий алгоритм разметки опорных вершин.

На рис. 1 приведена схема автомата, реализующего алгоритм в предположении, что автоматы W^0,W^1,W_i^2,W_i^3,W_i^4 реализуют пункты 0–4, где верхний индекс соответствует пункту, а нижний, если он есть, соответствующему типу опорной вершины. Таким образом остается проверить существование автоматов, реализующих каждый из пунктов 0–4.

Схема на рис. 1 фактически представляет из себя диаграмму Мура, в которой части, соответствующие разным автоматам, объеденены в блоки с некоторыми выделенными состояниями (вид автоматов

представляет верхняя часть рисунка), кроме того, стрелки из одного выделенного состояния блока в выделенное состояние другого блока означают передачу управления этому блоку без изменения положения автомата в графе, то есть в момент перехода к новому состоянию автомат стоит на месте.

Сигнальные состояния $q^{k1}\dots q^{kj}$ автомата W_i^4 в рассматриваемой схеме изображены как соответствующая пара q_0^{kl} и q_1^{kl}, такими, что все возможные переходы между основным блоком и q_0^{kl} и q_1^{kl} сгруппированы по принципу: все входящие в q_1^{kl}, а выходящие из q_0^{kl}. Стрелка из q_1^{kl} в q_0^{kl} означает переход без изменения положения автомата в графе. Такое раздвоение состояния дает возможность передать управление от одного автомата другому, а позже вернуть управление этому же автомату в том же состоянии. Состояния, помеченные крестиком, являются заключительными: если автомат попал в одно из них, значит он обошел граф.

5. Построение W^1,W_i^2,W_i^3,W^0

Данные автоматы не представляют сложности для построения, поэтому в явном виде будет построен только W^1. Для остальных автоматов будут даны алгоритмы их функционирования, которые в силу строго определенного объёма необходимой для работы информации, являются, как будет нетрудно видеть, автоматными.

Все приведенные алгоритмы будем рассматривать в том смысле, что работа автомата подчинена данному алгоритму: задан граф $G(V,E)$, система вращений R, алгоритм задает перемещение автомата по вершинам G, причем на каждом шаге автомат связан с некоторой вершиной $v \in V$ и некоторым ребром $e \in E$, выходящим из v. Алгоритм может задать перемещение автомата только по одному из ребер, выходящих из данной вершины, и покраску вершины одной нестираемой краской. Кроме того, алгоритму в каждый момент доступна только локальная информация о графе, а именно цвет вершины, где находится автомат, и количество ребер, из неё выходящих. Задание выходного ребра происходит так: алгоритм указывает

число y, меньшее степени текущей вершины $v(t)$, и автомат перемещается по ребру $(v(t),v') = R^y(e(t))$ в новую вершину $v(t+1) = v'$ с новым ребром $e(t+1) = (v', v(t))$. Также алгоритм может оставить автомат на месте, при этом вершина и ребро, с которыми он связан, не изменяются.

Определение. Назовем углом между ребрами $e_1 = (v,v_1)$ и $e_2 = (v,v_2)$, инцидентными вершине v, число $\theta \in \{0,1,\dots,deg(v)-1\}$, такое что $R^\theta(e_1) = e_2$ (при $e_1 = e_2$ $\theta = 0$), и обозначим его $\theta(e_1,e_2)$. Очевидно, что $\theta(e_2,e_1) = (deg(v) - \theta(e_1,e_2)) \bmod (deg(v))$.

Смысл введенного определения таков: если $e_1 = (v,v_1)$ и $e_2 = (v,v_2)$, и автомат, находясь в конфигурации (v,e_1), выберет в качестве выхода $\theta(e_1,e_2)$, то следующей вершиной будет v_2, а следующим ребром (v_2,v).

На наборах $\Theta_m = (\theta_1,\dots,\theta_m)$ равной длины, где $\theta_i \in \{0,1,\dots,k\}$, $k = \max\limits_{v\in G} deg(v)$, введем порядок $\Theta'_m > \Theta''_m$, если $\theta'_1 > \theta''_1$ или при $\theta'_i = \theta''_i$, $i < l$ и $\theta'_l > \theta''_l$.

5.1. Построение W^1

Построим автомат W_h^1, обходящий шар $B_h(v)$, $h > 0$. В частности, при $h = \varepsilon$ этот автомат будет W^1.

Алгоритм обхода $B_h(v)$ следующий.

Пусть автомат, находясь в (v,e), подал на выход последовательность углов $(\alpha_1,\dots,\alpha_m)$, попав в вершины $v_1,\dots,v_m$, причём $\alpha_1 < deg(v)$, $0 < \alpha_2 < deg(v_1),\dots,0 < \alpha_m < deg(v_{m-1})$. Если он может вернуться в (v,e) и выбрать другую последовательность $(\alpha_1,\dots,\alpha_m)$ и т.д., то, перебрав все возможные последовательности $(\alpha_1,\dots,\alpha_m)$, он обойдет $B_m(v)$ (так как любому пути длинны m соответствует некоторая последовательность $(\alpha_1,\dots,\alpha_m)$).

Вместо полного возвращения в (v,e), очевидно можно выбрать возвращение в v_l, $1 \leqslant l \leqslant m$ и рассмотреть другое продолжение $(\alpha_1,\dots,\alpha_{l-1},\alpha'_l,\dots,\alpha'_m)$, большее по порядку, если оно возможно. Таким образом, автомату достаточно помнить $(\alpha_1,\dots,\alpha_m)$, чтобы далее выбрать следующее возможное $(\alpha_1,\dots,\alpha_m)$.

Окончательно, алгоритм состоит в следующем.

1) Автомат пытается выбрать, пока возможно, углы $\alpha_1 = 1, \alpha_2 = 1, \ldots, \alpha_m = 1$ (то есть пока $m \leqslant h$ и $deg(v_i) > 1$ при $i \leqslant m-1$).

2) Пусть $\bar{\alpha} = (\alpha_1, \ldots, \alpha_m)$ выбрана и $m < h$, $deg(v_m) > 1$, и рассмотрим последовательности $\bar{\alpha}_1 = (\alpha_1, \ldots, \alpha_m, 1)$ $\bar{\alpha}_i = (\alpha_1, \ldots, \alpha_m, i)$, $(i < deg(v_m))$, тогда $\bar{\alpha}$ можно продолжить, если некоторая последовательность $\bar{\alpha}_i$ не была выбрана на каком-то из предыдущих шагов, причем i выбирается для продолжения минимальным (исходя из шага 3)).

3) Пусть $\bar{\alpha} = (\alpha_1, \ldots, \alpha_m)$ продолжить нельзя, то есть одно из условий 1) $m < h$, 2) $deg(v_m) > 1$, 3) существование i такого, что $\bar{\alpha}_i$ не выбиралось ранее, не выполнено. Тогда автомат выдает на выход 0 в первых двух случаях, 1 в третьем, запоминает α_m и переходит к рассмотрению $(\alpha_1, \ldots, \alpha_{m-1})$.

4) Пусть автомат обошёл все последовательности, начинающиеся на $\alpha_1 > 0$, то есть он, находясь в v, не может продолжить пустую последовательность последовательностью (α_1), так как все такие последовательности $(1), (2), \ldots, (deg(v) - 1)$ уже выбирались на предыдущих шагах. В этом случае автомат, выдав 1, переходит в v' и повторяет алгоритм 1)-3) для v', после чего, выдав 1, попадает в (v, e), уже обойдя $B_h(v)$.

Введем следующие вспомогательные множества:

$S = \{\uparrow, \downarrow\}, L = \{0, 1, \ldots, h\}, Z = \{0, 1, \ldots, k\}, Z^h = \underbrace{Z \times Z \times \ldots \times Z}_{h}$, $k \geqslant deg(v_i)$ при $v_i \in G$, h – радиус шара $B_h(v)$, S – множество, характеризующее направление движения автомата в шаре $B_h(v)$ – прямое или обратное, L – множество, элементы которого определяют длину пути, пройденного автоматом в прямом направлении, Z^h – множество, элементы которого характеризуют углы в вершинах пути, пройденного в прямом направлении. Более точно смысл данных множеств будет ясен из дальнейшего описания автомата.

Определим множество состояний автомата W_h^1 как $Q = S \times L \times Z^h \cup \{q_{end}\}$. Все состояния, кроме заключительного q_{end}, будут

иметь соответствующие разложения на компоненты, принадлежащие S, L, Z^h. В начальный момент для определенности задана конфигурация (v, e, q_0), где v – центр шара $B_h(v)$, e – начальное ребро и q_0 – начальное состояние автомата W_h^1. В процессе работы в момент t на вход автомата подается x_t – число ребер, инцидентных v_t, из текущей конфигурации (v_t, e_t, q_t), а на выход автомат выдает число y_t, меньшее $deg(v)$ или $\{-\}$, что интерпретируется как выбор ребра $R^{y_t}(e_t)$ и переход по нему в следующую вершину или сохранения v_t и e_t как элементов конфигурации в следующий момент времени $t+1$ с изменением лишь состояния автомата. Начальное состояние выберем $q_0 = (\uparrow, 0, (0, \ldots, 0))$.

Опишем функцию перехода состояний автомата $\varphi : X \times Q \to Q$, где X – множество возможных входов $X = \{1, \ldots, k\}$. Обозначим $q_t = (s(t), l(t), (z_0(t), \ldots, z_{l(t)-1}(t), 0, \ldots, 0))$. Будем считать, что в компоненте, соответствующей Z^h, ненулевыми могут быть только первые $l(t) - 1$ компонент, так как от компонент $l(t), \ldots, h$ поведение автомата не будет зависеть. Кроме того, функция переходов такова, что q_0 может эволюционировать только к состояниям подобного вида или q_{end}.

1) Если $(s(t) = \uparrow, (x_t = 1$ и $l(t) \geqslant 1)$ или $(l(t) = h)$, то $q(t+1) = (\downarrow, l(t) - 1, z_0(t), \ldots, z_{l(t)-1}, 0, \ldots, 0)$.

2) Если $(s(t) = \uparrow, (x_t > 1, l(t) \geqslant 1, l(t) < h)$, то $q(t+1) = (\uparrow, l(t) + 1, z_0(t), \ldots, z_{l(t)-1}, 1, 0, \ldots, 0)$.

3) Если $(s(t) = \downarrow, (x_t = z_{l(t)} + 1$ и $l(t) \geqslant 1)$, то $q(t+1) = (\downarrow, l(t) - 1, z_0(t), \ldots, z_{l(t)-1}, 0, \ldots, 0)$.

4) Если $(s(t) = \downarrow, (x_t \geqslant z_{l(t)} + 1$ и $l(t) \geqslant 1)$, то $q(t+1) = (\uparrow, l(t) + 1, z_0(t), \ldots, z_{l(t)} + 1, 0, \ldots, 0)$.

5) Если $(s(t) = \downarrow, (x_t > z_0(t), l(t) = 0)$, то $q(t+1) = (\uparrow, 1, z_0(t) + 1, 0, \ldots, 0)$.

6) Если $(s(t) = \downarrow, (x_t = z_0(t), l(t) = 0)$, то $q(t+1) = q_{end}$.

7) Если $(s(t) = \uparrow, (l(t) = 0)$, то $q(t+1) = (\uparrow, 1, 1, 0, \ldots, 0)$.

Функция выходов $\psi : X \times Q \to \{0, 1, \ldots, k-1, \{-\}\}$ описывается так:

1) Если $l(t) = h$, то $\psi(x(t), q(t)) = 0$.

2) Если $l(t) = 0$, $s(t) = \downarrow$, $x(t) = z_0(t)$, то $\psi(x(t), q(t)) = \{-\}$.

3) В остальных случаях $\psi(x(t), q(t)) = 1$.

5.2. Построение W_i^2

Алгоритм раскраски опорной вершины будет основан на том, что существует путь $v, v_1, v_2, \ldots, v_m$ достаточной длины, что $v_i \in O_i(v)$. Очевидно, что если автомат разметит v, v_2, v_3, то получится вершина типа 1, v, v_2, v_3, v_4 – вершина типа 2 и v, v_2, v_3, v_4, v_5 – вершина типа 3, кроме того, от v_5 до v ближе, чем до любой другой опорной вершины, уже размеченной в графе.

Алгоритм состоит в следующем (приведём для W_3^2):

1) автомат метит v.

2) автомат последовательно выбирает возможные наборы углов $\alpha_1, \ldots, \alpha_5$, проходя вершины $v_1, \ldots, v_5$, где $\alpha_1 < deg(v)$, $0 < \alpha_i < deg(v_{i-1})$.

3) В вершине v_5 автомат обходит $B_4(v_5)$. Если v оказалась в $B_4(v_5)$, значит $v_5 \notin O_5(v)$ и надо искать следующую v_5, попробовав следующую последовательность углов. Проверить вершину из $B_4(v_5)$ на то, что она является v можно так: а) она помечена; б) в её $O_1(v)$ нет помеченных. Так как она ближе к v_5, чем любая другая опорная, то в $B_4(v_5)$ может попасть помеченная вершина, не являющаяся v, только если одна из её соседних вершин также помечена.

4) Если нашлась вершина v_5, что $B_4(v_5)$ не содержит v, значит $v_5 \in O_5(v)$. Далее автомат метит v_5, v_4, v_3, v_2, возвращаясь по пути $(0, deg(v_4) - \alpha_5, deg(v_3) - \alpha_4, deg(v_2) - \alpha_3, deg(v_1) - \alpha_2, deg(v) - \alpha_1, 0)$, то есть попадая в конфигурацию (v, e) с тем же, что и в начале, ребром и вершиной. При этом, очевидно, v становится размеченой, как опорная вершина третьего типа.

Аналогично строятся W_1^2 и W_2^2.

5.3. Построение W_i^3

Построим автомат для $i = 1$, остальные строятся аналогично.

Автомат должен проверить фактически следующие условия:

1) В $B_\varepsilon(v)$ нет вершины типа 2.

2) В $B_{\varepsilon-1}(v)$ нет опорных вершин.

3) В $O_\varepsilon(v)$ есть опорная вершина типа 3.

Будем считать, автомат начал работу в конфигурации (v, e, q_0). Условие 3) можно не проверять, так как представленный кандидат будет заведомо не далее ε от вершины типа 3, а то что он находится от неё не меньше чем на ε гарантировано условием 2).

Алгоритм для W_1^3 состоит в следующем: автомат, как и в случае обхода $B_\varepsilon(v)$, будет порождать всевозможные последовательности углов $\alpha_1, \ldots, \alpha_n$, $n \leqslant \varepsilon$ и в каждой конечной вершине последовательности будет проверять, является ли эта вершина опорной. При $n < \varepsilon$ в случае обнаружения опорной вершины автомат вернётся в начальную вершину и примет конфигурацию (v, e, q_n). При $n = \varepsilon$ в случае обнаружения опорной вершины типа 2, автомат так же перейдет в конфигурацию (v, e, q_n). Если же при обходе $B_{\varepsilon-1}$ не найдется опорных вершин и в $O_\varepsilon(v)$ не найдется опорных вершин типа 2, то автомат вернется в конфигурацию (v, e, q_y).

Алгоритм проверки вершины v^0 на то, что она опорная конкретного типа, следующий:

1) Автомат проверяет, что v^0 помеченная.

2) Автомат проверяет, что $O_1(v^0)$ не помечено.

3) Автомат проверяет, что $O_2(v^0)$ содержит помеченную вершину.

4) Автомат считает длину цепочки из помеченных вершин, первая из которых находится в $O_2(v^0)$.

Первые три пункта проверяются очевидным образом. Последний пункт проверяется так: автомат находит помеченную вершину v_2 в $O_2(v^0)$, далее, обойдя $O_1(v_2)$, автомат находит помеченную вершину v_3 в $O_3(v^0)$. Если v^0 вершина типа более первого, то в $O_1(v_3) \setminus v_2$, он найдет помеченную $v_4 \in O_4(v^0)$, а если v^0 опорная третьего типа,

то в $O_1(v_4) \setminus v_3$ найдется и помеченная вершина $v_5 \in O_5(v^0)$. То есть автомату достаточно последовательно обойти $v_2, \ldots, v_n$ и по номеру последней найденной установить тип вершины. Алгоритм работает в предположении, что между уже построенными опорными вершинами расстояние не менее ε, что дает гарантию того, что, если вершина v^0 опорная, то в $B_6(v^0)$ не попадет помеченная из разметки другой опорной вершины.

5.4. Построение W_0

Автомат W_0 действует следующим образом:

В начальный момент автомат находится в конфигурации (v_0, e, q_0). На первом шаге автомат метит вершину v_0. Далее автомат порождает последовательности углов $\alpha_1, \alpha_2, \alpha_3$ и в конечной вершине v_3 проверяет, есть ли в $B_2(v_3)$ помеченная. Если находится такая вершина v_3, что $O_2(v_3)$ не содержит помеченных, то $v_3 \in O_3(v_0)$ и соответственно $O_3(v_0)$ не пусто. Найдя такую вершину, автомат переходит в конфигурацию (v_0, e, q_n). Если же автомат такой вершины не находит, он переходит в конфигурацию (v_0, e, q_n). Заметим, что автомат, размечающий $B_3(v_0)$, должен воспользоваться тем, что v_0 уже размечена.

6. Алгоритм автоматного обхода опорных вершин, размеченных на некотором шаге, но без использования краски

Будем считать, что некоторая часть графа раскрашена в соответствии с алгоритмом разметки опорных вершин и при этом работа алгоритма остановлена сразу после разметки некоторой опорной вершины из очередного S^{i+1}. Здесь будет рассмотрен алгоритм обхода автоматом всех размеченных вершин из $S^1, \ldots, S^{i+1}$, при этом автомату доступна информация только о цвете текущей вершины графа G и число ребер из нее выходящих. Кроме того, автомат может выбирать следующую вершину, указывая поворот относительно ребра, по которому он пришёл, не имея информации о самом ребре (то есть функционирует как автомат в r-лабиринте с вершинами разных цветов).

Для описания алгоритма введем несколько определений.

Рассмотрим опорную вершину v_0 с соответствующей разметкой в $B_5(v_0)$. Пусть $e_m = (v_2, v_3)$, где v_2, v_3 – помеченные вершины и $v_2 \in O_1(v_0), v_3 \in O_1(v_0)$.

Рассмотрим все возможные пути $L^i = (v_2, v_1^i, v_0)$, где $(v_2, v_1^i) \in E$, $(v_1^i, v_0) \in E$ и рассмотрим последовательность углов $\Theta^i = (\theta(e_m, (v_2, v_1^i))\theta((v_1^i, v_2), (v_1^i, v_0)))$ из однозначного соответствия v^i и Θ^i и неравентсва Θ^i с Θ^j при $i \neq j$ следует существование v_1^I, такого что $\Theta^I < \Theta^i$ для любого $i \neq I$.

Определение. Ребро (v_0, v_1^I) назовем опорным для опорной вершины v_0 и обозначим его $e_0 = e_0(v_0)$.

Рассмотрим v_0 – опорную вершину и некоторый путь $\Pi = (v_0, v_1, \ldots, v_m)$ длины m, начинающийся в v_0 и заканчивающийся в v_m. Характеристическим набором или просто характеристикой пути Π относительно v_0 назовем набор $(\theta_0, \theta_1, \ldots, \theta_{m-1})$, где $\theta_0 = \theta(e_0, (v_0, v_1)), \theta_1 = \theta((v_0, v_1), (v_1, v_2)), \ldots, \theta_{m-1} = \theta((v_{m-2}, v_{m-1}), (v_{m-1}, v_m))$.

Пусть $v_0 \in S^{i+1}$, тогда назовем предком v_0 вершину v_p из S^i, такую что $L(v_0, v_p) = \varepsilon$ и существует путь Π длины ε между v_0 и v_p, что его характеристика минимальна среди всех путей длины ε, соединяющих v_0 и опорные вершины из S^i. Этот путь назовем трассой v_0. Путь Π^{-1} из v_p в v_0, обратный Π, назовём обратной трассой и его характеристику относительно v_p назовем обратной характеристикой трассы для v_0.

Теперь построим граф $G_1(V_1, E_1)$, устроенный следующим образом:

Каждой опорной вершине $G(V, E)$ будет соответствовать вершина графа G_1 с пометкой, соответствующей типу данной опорной вершины. Каждой прямой и обратной трассе в G будет соответствовать ребро из E_1 с пометкой в виде соответствующей характеристики

(прямой или обратной).

Заметим, что если рассматривать G как неориентированный граф (из существования ребра, обратного данному), то полученный граф является деревом, так как 1) граф связный, 2) число трасс равно $|V_1| - 1$ (опорная вершина из S^1 трассы не имеет, а остальные имеют ровно одну трассу). Кроме того, если из вершины исходит несколько обратных трасс, то все они имеют разные пометки, к тому же отличные от пометки прямой трассы. Еще одним свойством является то, что если трасса начинается в вершине типа 3, то заканчивается в вершине типа 2, и, соответственно, начинается в типа 2 и заканчивается в типа 1, начинается в вершине типа 1 и заканчивается в вершине типа 3. В вершине v_0 из S^1 (назовем ее корнем) вообще не начинается никакая трасса, кроме того, в её $O_\varepsilon(v_0)$ находятся опорные вершины лишь второго типа, и по этой причине она легко может быть вычислена.

Рассмотрим алгоритм обхода графа G_1 при условии, что начинается обход в произвольной вершине из V_1.

1) Спуск в корень. Автомат выбирает трассу текущей опорной вершины, переходя к следующей. Так продолжается, пока автомат не попадает в опорную вершину без трассы, то есть в корень.

2) Прямой ход. Автомат в текущей опорной вершине выбирает обратную трассу к трассе, заканчивающейся в данной вершине, с минимальной характеристикой (так как обратные характеристики все разные, то их можно упорядочить), после чего переходит в опорную вершину, соответствующую этой обратной трассе. Так автомат продолжает до тех пор, пока не попадёт в вершину, не имеющую трассы, которая в ней заканчивается. Здесь автомат начинает обратный ход.

3) Обратный ход. Автомат выбирает трассу и переходит по ней в следующую опорную вершину, пока не попадет в такую вершину, где есть трассы, в ней заканчивающиеся, с обратной характеристикой, большей чем обратная характеристика трассы, по которой он пришел. В этом случае автомат выбирает следующую за ней обратную трассу и начинает прямой ход. Если же автомат попал в корень по трассе, обратная к которой имеет наибольшую характеристику, то

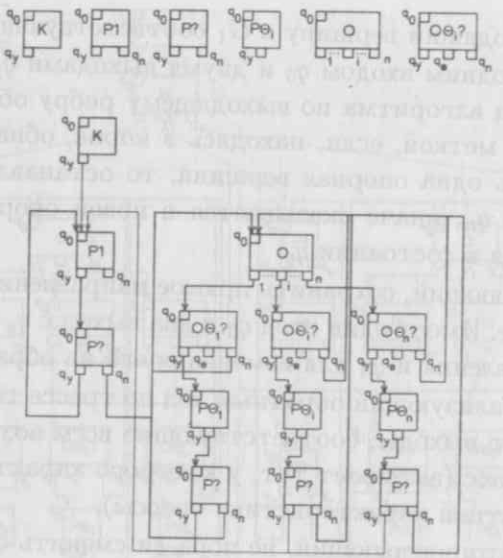

Рис. 2. Автомат исполняющий алгоритм обхода размеченных опорных вершин в графе G_1

автомат останавливается. После остановки автомат, очевидно, уже обошел G_1.

Данный алгоритм, очевидно, реализуем автоматом, если ему a) доступны метки вершины, метки ребер, выходящих из вершины; b) если он может выбирать ребро, выдав его метку и c) зная метку ребра, по которому он прошел, определить ему противоположное.

Диаграмму Мура (рис. 2) для автомата D, обходящего вершины G_1, схематически можно представить в виде комбинации блоков P_1,$P\Theta_i?$,O,$O\Theta_i$,$P?$.

Как и в случае рис. 1, под стрелкой из состояния одного автомата в состояние другого автомата понимается передача управления без изменения текущей вершины и ребра в лабиринте G.

Каждый из блоков реализует следующие функции.

K – блок с одним входом q_0 и выходом q_y, спускающийся в корень

G_1, то есть находящий вершину в G_1 соответствующую корню в G.

P_1 – блок с одним входом q_0 и двумя выходами q_y,q_n, реализующий прямой ход алгоритма по выходящему ребру обратной трассы с минимальной меткой, если, находясь в корне, обнаруживает, что размечена лишь одна опорная вершина, то останавливается в корне в состоянии q_n, иначе оказывается в новой опорной вершине и останавливается в состоянии q_y.

$P?$ – проверяющий, сохранить прямое направление или изменить его на обратное. Имеет один вход q_0 и два выхода: q_y для сохранения прямого направления и q_n для изменения его на обратное.

O – блок, реализующий обратный ход по трассе текущей опорной вершины, имеет выходы, соответствующие всем возможным характеристикам трасс (выбирает тот, у которого характеристика соответствует обратной характеристике трассы).

$O\Theta_i?$ – блок, проверяющий, не пора ли сменить обратное направление на прямое (с учетом обратной характеристики трассы, по которой автомат только что прошёл) или не окончен ли алгоритм. Имеет вход q_0 и выходы а) q_y в случае перехода к прямому направлению алгоритма, б) q_n в случае повторения обратного хода алгоритма, в) q_e в случае, когда автомат завершает свою работу.

$P\Theta_i$ – то же, что и P_1, но по ребру, следующему за Θ_i, то есть переход по обратной трассе, заканчивающейся в данной опорной вершине, следующей по порядку за обратной трассой, по которой автомат попал в данную вершину на обратном ходу.

Таким образом, для того чтобы реализовать алгоритм обхода опорных вершин в G, достаточно показать, что существует автомат, способный собрать необходимую информацию в конкретной опорной вершине и передать ее автомату, действующему согласно приведенному выше алгоритму. Более того, этот автомат должен, обойдя некоторую окрестность данной опорной вершины v_0, вернуться в нее и оказаться в таком состоянии, которое описывает:

1) характеристику трассы v_0 (или то, что это корень);

2) обратную характеристику трассы v_0;

3) характеристики трасс, заканчивающихся в v_0;

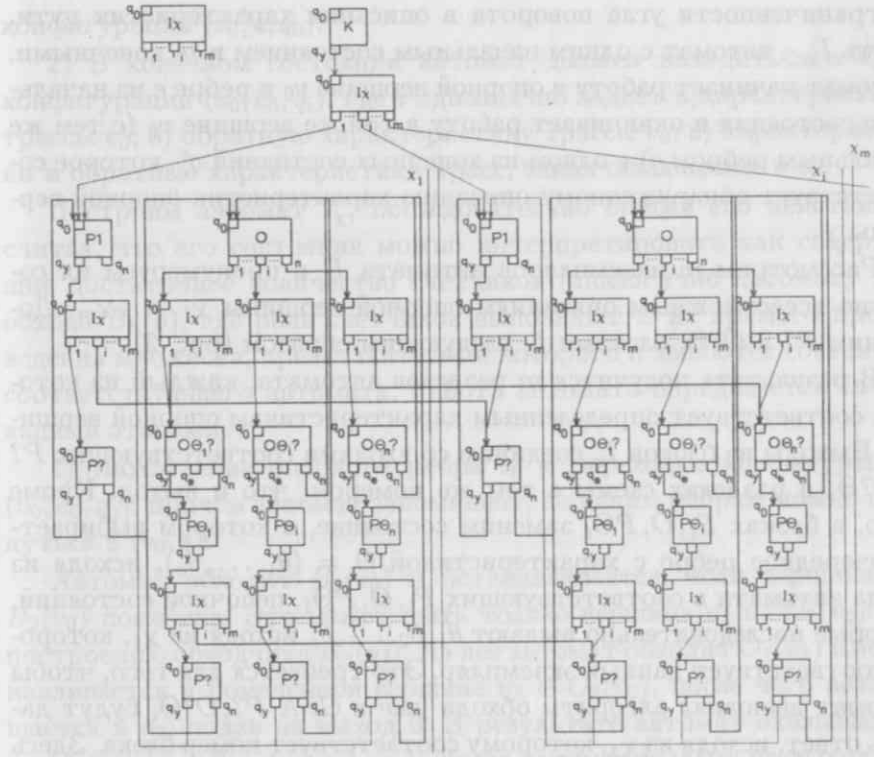

Рис. 3. Автомат исполняющий алгоритм обхода размеченных опорных вершин в графе G

4) обратные характеристики трасс, заканчивающихся в v_0.

Если такой автомат получен, то автомат, реализующий алгоритм обхода уже построенных опорных вершин, конструируется следующим образом (рис. 3). Для получения автомата, реализующего алгоритм в G, рассмотрим следующую схему.

Пусть $\chi_1, \ldots, \chi_m$ – все возможные описания, характеризующие опорную вершину с точки зрения характеристик трассы самой вершины и характеристик трасс, заканчивающихся в ней. Их конечное число, что следует из ограниченности степени вершин графов в G_k^R

и ограниченности угла поворота в описании характеристик пути. Пусть I_χ – автомат с одним начальным состоянием и m конечными. Автомат начинает работу в опорной вершине v_0 и ребре e из начального состояния и оканчивает работу в той же вершине v_0 (с тем же начальным ребром e) в одном из конечных состояний q_i^e, которое соответствует обнаруженному описанию характеристик опорной вершины χ_i.

Рассмотрим m экземпляров автомата D и пронумеруем их согласно всевозможным описаниям опорной вершины $\chi_1, \ldots, \chi_m$. Дополним $P?$ и $O?\Theta_i$ блоками I_χ следующим образом (рис. 3).

В результате получится m разделов автомата, каждый из которых соответствует определенным характеристикам опорной вершины. Выходы из блоков I_χ соединим со входами соответствующих $P?$ и $O?\Theta_i$ в разделах схемы с тем же номером, что и выход. Кроме того, в блоках $P_1, O, P\Theta_i$ заменим состояние, в котором выбирается очередное ребро с характеристикой $\Theta = (\theta_1, \ldots, \theta_\varepsilon)$, исходя из входа автомата в соответствующих $P_1, O, P\Theta_i$, цепочкой состояний, которые последовательно выдают $\theta_1, \theta_2, \ldots, \theta_\varepsilon$ исходя из χ_j, которому соответствует данный экземпляр. Это требуется для того, чтобы автомат выполнял алгоритм обхода уже в G. А $P?, O?\Theta_i$ будут давать ответ, исходя из χ_j, которому соответствует номер блока. Здесь все элементарные блоки пользуются информацией о текущей опорной вершине (то есть для каждого раздела каждый элементарный блок свой).

Конструкция всех элементарных блоков $O, P, P1, P\Theta_i, O?\Theta_i, P?$, кроме I_χ и K, очевидна. Для блока K построение аналогично построению части I_χ (достаточно выбирать трассу у опорной вершины, пока она есть). Таким образом, остается установить существование I_χ.

7. Построение автомата I_χ

Автомат должен выполнять следующие функции.

1) В начальном состоянии он находится в опорной вершине v_0 и конфигурации (v_0, e, q_0).

2) В конечном состоянии автомат должен находиться в v_0 и конфигурации (v_0, e_0, q_i), где i однозначно задаёт а)характеристику трассы v_0; б) обратную характеристику трассы v_0; в) характеристики и обратные характеристики трасс, заканчивающихся в v_0.

Построим автомат I_χ, последовательно описав его действия и считая, что его состояния можно интерпретировать как содержащие достаточное количество счетчиков (аналогично автомату для обхода $B_h(v)$, где роль счетчиков выполняют Z из прямого произведения множеств, представителями которого и являются состояния соответствующего автомата. Работа автомата определяется следующими этапами.

1) Нахождение опорного ребра e_0 и переход в конфигурацию (v_0, e_0, q_0'), причем автомат запоминает, по каким ребрам можно вернуться в (v_0, e).

Автомат обходит $B_2(v_0)$ и останавливается, если вершина из $B_2(v_0)$ помечена. Это может быть только вершина v_2 из $O_2(v_0)$ (по построению обходчика $B_h(v)$). Из неё автомат обходит $O_1(v_2)$ и останавливается в помеченной вершине $v_3 \in O_3(v_0)$, после чего возвращается в v_2, подав на выход 0. В результате автомат оказывается в кофигурации $(v_2, (v_2, v_3), q_K)$. Далее автомат выдает на выход числа γ_1, γ_2 где $0 < \gamma_1 < deg(v_2)$, $0 < \gamma_2$ и меньше степени вершины, куда пришел автомат, подав γ_1. Далее, если после прохода v'' автомат попал в помеченную вершину v' (если v' не помечена то автомат выдает $0, deg(v'') - \gamma_2$), то он проверяет её окрестность на наличие помеченных вершин, подавая на выход числа γ_3, $0 < \gamma_3 \leqslant deg(v')$ и 0 после очередного γ_3. Если среди вершин, которые он обошёл в $O_1(v')$, были помеченные, то автомат выдает числа 0 и $deg(v'') - \gamma_2$, оказываясь в v_2. Далее пробуются следующие γ_1, γ_2. Если же в v' $O_1(v')$ не помечена, при помеченной v', то автомат попал в v_0 и находится в конфигурации $(v_0, e_0, q_{\gamma_1,\gamma_2})$, выдав $\{-\}$ и перейдя в q_0', автомат выполнит первый этап работы.

2) Обход O_ε и нахождение трассы для v_0.

Автомат последовательно подает на выход $\gamma_1, \gamma_2, \ldots, \gamma_\varepsilon$, попа-

дая в вершины $v_1, \ldots, v_\varepsilon$, причем $0 \leqslant \gamma_1 < deg(v_0), 0 < \gamma_2 < deg(v_1), \ldots, 0 < \gamma_\varepsilon < deg(v_{\varepsilon-1})$. Кроме того, автомат запоминает в своих счетчиках степени $deg(v_i)$. Далее, если v_ε – опорная типа, предыдущего к типу v_0 (то есть $T(v_0) \bmod 3 = T(v_\varepsilon) + 1 \bmod 3$), проверку можно осуществить так, что автомат после неё окажется в конфигурации с тем же ребром e_ε и той же вершиной v_ε. Если v_ε не подходит, то автомат, возвращаясь в v_0, переходит в конфигурацию с ребром e_0 и пробует следующий набор $\gamma_1, \ldots, \gamma_\varepsilon$. Если же v_ε нужного типа, то $\gamma_1, \ldots, \gamma_\varepsilon$ – характеристика трассы. Если все наборы $\gamma_1, \ldots, \gamma_\varepsilon$ перебраны и трассы не нашлось, значит v_0 является корнем.

3) Нахождение обратной характеристики трассы.

Автомат находится в вершине v_ε и конфигурация содержит ребро e, по которому автомат попал в v_ε, пройдя путь по вершинам $v_1, \ldots, v_{\varepsilon-1}$ и углам $\gamma_1, \ldots, \gamma_\varepsilon$. Очевидно, обратная характеристика трассы будет иметь $\gamma'_2 = deg(v_{\varepsilon-1}) - \gamma_\varepsilon, \ldots, \gamma'_\varepsilon = deg(v_1) - \gamma_2$. Вопрос открыт только для угла γ'_1 между e_ε и $(v_\varepsilon, v_{\varepsilon-1})$. Чтобы найти этот угол, автомат выполняет следующую процедуру. Последовательно выбирает углы $\alpha_1\alpha_2\alpha_3$, попадая в вершины $v'v''v'''$, где $0 \leqslant \alpha_1 < deg(v_\varepsilon), 0 < \alpha_2 < deg(v'), 0 \leqslant \alpha_3 < deg(v'')$. Если хотя бы одна из v'', v''' не помечена, автомат возвращается и пробует следующую возможную комбинацию $\alpha_1\alpha_2\alpha_3$. Если же v'' и v''' помечены, то автомат выбирает углы $0, \beta_1, \beta_2$ и проходит вершины v''', v_β, v_α, где $0 < \beta_1 < deg(v'''), 0 < \beta_2 < deg(v_\beta)$. Далее, если v_α помечена, автомат проверяет, есть ли в окрестности v_α помеченные вершины, если нет, то $v_\alpha = v_\varepsilon$. Если $v_\alpha \neq v_\varepsilon$, автомат пробует следующие β_1, β_2, пока не найдутся такие β_1, β_2, что $v_\alpha = v_\varepsilon$. В этом случае, если $\alpha_3 + \beta_1 = deg(v'')$, то и $\theta(e_\varepsilon, (v_\varepsilon, v_{\varepsilon-1})) = deg(v_\varepsilon) - \alpha_1$, если же $\alpha_3 + \beta_1 < deg(v'')$, то автомат пробует следующие $\alpha_1\alpha_2\alpha_3$ и т.д., пока не будет найдено $\alpha_3 + \beta_1 = deg(v'')$ и тем самым не будет установлена характеристика обратной трассы. После этого автомат может вернуться в v_0 в конфигурацию (v_0, e_0, q^t), где q^t «помнит» характеристики трассы прямую и обратную.

4) Нахождение характеристик и обратных характеристик трасс, заканчивающихся в v_0.

Автомат, последовательно генерируя $\gamma_1, \ldots, \gamma_\varepsilon$ с условием из 2), проверяет, является ли последняя вершина опорной подходящего типа, и, если да, то аналогично 3) находит обратную характеристику для пути $v_0, \ldots, v_\varepsilon$ и сравнивает её с характеристикой трассы v_ε. Если они совпадают, то автомат запоминает соответствующие обратную и прямую характеристики трассы и пробует новый вариант $\gamma_1, \ldots, \gamma_\varepsilon$. После перебора всех возможных $\gamma_1, \ldots, \gamma_\varepsilon$ автомат собирает требуемую информацию о характеристиках и обратных характеристиках трасс, заканчивающихся в v_0. Затем автомат переходит в (v_0, e, q^χ), где q^χ описывает все требуемые параметры v_0.

8. Построение W_i^4

Так как автомат, обходящий окрашенную часть, уже построен, остаётся понять, как добавить к нему возможность представлять новых кандидатов на присоединение к очередному S^i. Это делается так: автомат обходит каждый шар $B_\varepsilon(v)$, для каждой опорной вершины v типа, предыдущего к i, и каждое состояние, в которое он при этом попадает, объявляется сигнальным (или для сокращения числа кандидатов можно взять лишь те состояния, в которых может оказаться автомат, находясь в $O_\varepsilon(v)$).

На этом построение $A^k(X', Y', Q_a, \varphi_a, \psi_a, q_a^0)$ окончено.

Замечание 1. Если граф уже раскрашен целиком согласно алгоритму, то он обходится автоматом W_i^4, при любом i, и очевидно, без использования краски. Таким образом существование $B^k(X', Y, Q_b, \varphi_b, \psi_b, q_b^0)$ доказано.

Замечание 2. Если рассматривать бесконечные графы (связные, со счетным числом вершин, с ограничением на степень вершины и со структурой r-лабиринта), то, как нетрудно видеть, теорема 2 верна (в силу существования некоторого конечного пути из начальной вершины в произвольную). А так как при реализации автоматом данного алгоритма конечность графа не существенна, то теорема 1 справедлива и в случае бесконечных r-лабиринтов, с соответствую-

щим понятием обхода.

Список литературы

[1] Budach L. Automata and labyrinth. Math. Nachrichten 86, 1978, 195–282.

[2] Hemmerling A. Labyrinth problems. Labyrinth-searching abilities of automata. Teubner-Texte zur Mathematik, v 114, Leipzig, 1989.

[3] Насыров А.З. Об обходе лабиринтов автоматами, оставляющими нестираемые метки. Дискретная математика. 1997, т. 9, №1, 123–133.

[4] Кудрявцев В.Б., Алёшин С.В., Подколзин А.С. Введение в теорию автоматов. М.: Наука, 1985.

[5] Кудрявцев В.Б., Ушчумлич Щ., Килибарда Г. О поведении автоматов в лабиринтах. Дискретная математика. 1992, т. 4, №3, 3–28.

[6] Килибарда Г. Новое доказательство теоремы Будаха-Подколзина. Дискретная математика. 1991, т. 3, №3, 135–146.

[7] Золотых А.А. Обход лабиринтов с ограниченными в фиксированных направлениях дырами. Дискретная математика. 1993, т. 5, №1, 59–69.

[8] Харари Ф. Теория графов. Пер. с англ. М.: Мир, 1973.

[9] Будах Л. Автоматы в лабиринтах. Пробл. кибернетики, 34, 1978, 83–74.

[10] Кудрявцев В.Б., Подколзин А.С., Ушчумлич Ш. Введение в теорию абстрактных автоматов. М.: изд-во Московского ун-та, 1985.

Об отличимости инициальных автоматных лабиринтов конечными автоматами

И.С. Грунский, Р.И. Олейник

Исследуется проблема распознавания неизвестных характеристик автоматных лабиринтов конечным автоматом, блужающим по этим лабиринтам. Рассмотрены условия различения двух лабиринтов автоматом. Показано, что любые два неизоморфные лабиринта различаются экспериментом кратности не больше двух. Найдены точные верхние оценки наименьшего времени такого различения. Описан алгоритм построения таких экспериментов.

1. Введение

Рассматривается задача об отличимости автоматных лабиринтов конечным автоматом. Автоматные лабиринты представляют собой конечные орграфы переходов инициальных автоматов без выхода [1] и конечных отмеченных систем [2]. С помощью таких графов можно описывать различные ситуации [1] и динамику взаимодействия систем [2]. Конечный автомат стартует в начальной вершине одного из двух орграфов, блуждает по нему (при этом может оставлять отметки в вершинах этого орграфа), и за конечное время автомат должен выдать на выходе информацию, в каком именно графе он блуждал.

В работе устанавливаются точные верхние оценки наименьшего времени, за которое различаются два автоматных лабиринта. В отличие от [1], рассматриваются не только простые (однократные), но