ИПМ им.М.В.Келдыша РАН

Абрау-2018 • Труды конференции


И.Б. Бурдонов, А.С. Косачев

# Настройка распределенной сети на кратчайшие пути

***Рекомендуемая форма библиографической ссылки***

Бурдонов И.Б., Косачев А.С. Настройка распределенной сети на кратчайшие пути // Научный сервис в сети Интернет: труды XX Всероссийской научной конференции (17-22 сентября 2018 г., г. Новороссийск). — М.: ИПМ им. М.В.Келдыша, 2018. — С. 89-99. — URL: http://keldysh.ru/abrau/2018/theses/3.pdf doi:10.20948/abrau-2018-3

***Размещена также презентация к докладу***

# Настройка распределенной сети на кратчайшие пути

**И.Б. Бурдонов**[1], **А.С. Косачев**[1]

[1] *Институт системного программирования им. В.П. Иванникова РАН (ИСПРАН)*

**Аннотация.** Рассматривается распределенная сеть, в основе которой лежит сильно связный ориентированный нумерованный корневой граф без кратных ребер и петель, в котором дуги, исходящие из вершины, перенумерованы от 1 до полустепени исхода вершины. Ставится задача обеспечения передачи сообщений из каждой вершины в каждую по кратчайшим путям. Для этого предлагается построить в каждой вершине $a$ графа отображение, которое каждой вершине с номером $b$ ставит в соответствие номер дуги, с которой начинается кратчайший путь из $a$ в $b$. Такое отображение полезно для решения многих задач, в частности, задачи самонастройки программно-конфигурируемой сети. Сетевой коммутатор (switch) хранит в программно-настраиваемых таблицах отображение адреса целевого хоста в номер выходного порта. Время передачи пакетов минимально, если в отображении использованы кратчайшие пути. Другая задача – симуляция распределенных алгоритмов, разработанных для сетей с неориентированным графом связей, на классе сетей с ориентированным графом связей. Здесь приходится решать проблему отката (backtracking problem): вместо передачи сообщения из конца дуги $a \to b$ в ее начало сообщение передается по ориентированному пути $b \to ... \to a$. Предполагается, что алгоритм построения отображения должен начинаться и заканчиваться в корне графа. Предлагаются два алгоритма построения отображения: «быстрый» алгоритм с оценками $T=\boldsymbol{O}(n)$ и $N=\boldsymbol{O}(n)$ и «экономный» алгоритм с оценками $T=\boldsymbol{O}(n^2)$ и $N=\boldsymbol{O}(1)$, где $T$ – время (в тактах) работы алгоритма, $N$ – число сообщений, одновременно передаваемых по дуге, $n$ – число вершин графа. Доказано, что эти оценки являются нижними оценками на классе алгоритмов, которые правильно работают на классе всех рассматриваемых графов.

**Ключевые слова:** ориентированный граф; корневой граф; нумерованный граф; распределенные алгоритмы; задачи на графах; кратчайшие пути.

# Configuring a Distributed Network on the Shortest Paths

**I.B. Burdonov**[1], **A.S. Kossatchev**[1]

[1] *Ivannikov Institute for System Programming of the RAS (ISPRAS)*

**Abstract.** We consider a distributed system based on a strongly connected directed numbered rooted graph without multiple edges and loops in which the arcs outgoing from the vertex are numbered from 1 to the out-degree of this vertex. The task is to ensure the message passing from each sender to each receiver by the shortest paths. For this it is proposed to construct at each vertex *a* of the graph a mapping that assigns to each vertex number *b* the arc number with which the shortest path starts from *a* to *b*. This mapping is useful for solving many problems, in particular, the task of self-configuration a program-configurable network. The network switch stores in program-configurable tables the mapping of the address of the target host to the number of the output port. The time for passing packets is minimal, if the shortest paths are used in the mapping. Another problem is the simulation of distributed algorithms developed for distributed system based on an undirected connection graph on a class of systems based on a directed connection graph. Here we have to solve the backtracking problem: instead of passing a message from the end of the arc $a \to b$ to its beginning, the message is passing along the directed path $b \to \ldots \to a$. It is assumed that the mapping construction algorithm should start and end at the root of the graph. Two algorithms for constructing a mapping are proposed: a "fast" algorithm with estimates $T = O(n)$ and $N = O(n)$ and an "economical" algorithm with estimates $T = O(n^2)$ and $N = O(1)$, where $T$ is the algorithm running time, $N$ is the number of messages simultaneously transmitted along the arc, $n$ is the number of vertices of the graph. It is proved that these estimates are lower bounds on the class of algorithms that correctly operate on the class of all graphs under consideration.

**Keywords:** directed graph; rooted graph; numbered graph; distributed algorithms; graph problems; shortest paths.

## 1. Введение

Распределенная система – это граф, в вершинах которого располагаются вычислительные единицы (автоматы), обменивающиеся между собой сообщениями, посылаемыми по ребрам графа. В ориентированном графе ребро имеет направление от одного своего конца (исходит из него) к другому своему концу (входит в него), называется дугой и соответствует симплексному каналу передачи сообщений от начала дуги к ее концу.

Предполагается, что граф: 1) сильно связный (из каждой вершины можно попасть в каждую вершину, двигаясь по дугам), 2) *нумерованный* – вершинам присвоены номера от 1 до *n*, 3) *корневой* – выполнение алгоритма начинается и

заканчивается в выделенной вершине – корне графа, 4) *упорядоченный* – дуги, исходящие из вершины, перенумерованы от 1 до полустепени исхода вершины (число дуг, исходящих из вершины). Упорядоченность графа нужна для того, чтобы автомат в вершине мог указать исходящую дугу, по которой посылается сообщение.

Для краткости вместо «автомат в вершине» будем говорить просто «вершина». Память вершины понимается как набор *переменных*, а сообщение – как набор *параметров*.

Сообщение двигается по дуге не дольше 1 такта. В одном срабатывании автомат принимает одно сообщение по одной входящей дуге и посылает по каждой исходящей дуге ограниченное число (в предлагаемых алгоритмах не более трех) сообщений. Для оценки будем пренебрегать временем срабатывания автомата, т.е. сложность алгоритмов оценивается как время перемещения сообщений по дугам.

Будут даны следующие оценки алгоритмов: $T$ – время работы, $A$ – размер памяти вершины как сумма размеров (в битах) переменных, $M$ – размер сообщения как сумма размеров (в битах) его параметров, $N$ – число сообщений, одновременно передаваемых по одной дуге.

Для того чтобы в графе с $n$ вершинами из любой вершины можно было попасть в любую вершину по кратчайшему пути (пути минимальной длины), ставится задача построения в каждой вершине $a$ отображения $f_a : [1..n] \to [0..d^-(a)]$, где $d^-(a)$ – полустепень исхода вершины $a$, которое каждой вершине с номером $b$ ставит в соответствие номер дуги, с которой начинается кратчайший путь из $a$ в $b$, если $a \neq b$, или 0 в противном случае. Размер такого отображения равен $\boldsymbol{O}(n\log d^-(a)) = \boldsymbol{O}(n\log n)$, поэтому $A = \Omega(n\log n$, что на порядок меньше памяти для хранения всего упорядоченного графа $\boldsymbol{O}(n^2 \log n)$ [1].

Результаты работы показаны наглядно на рис. 1.

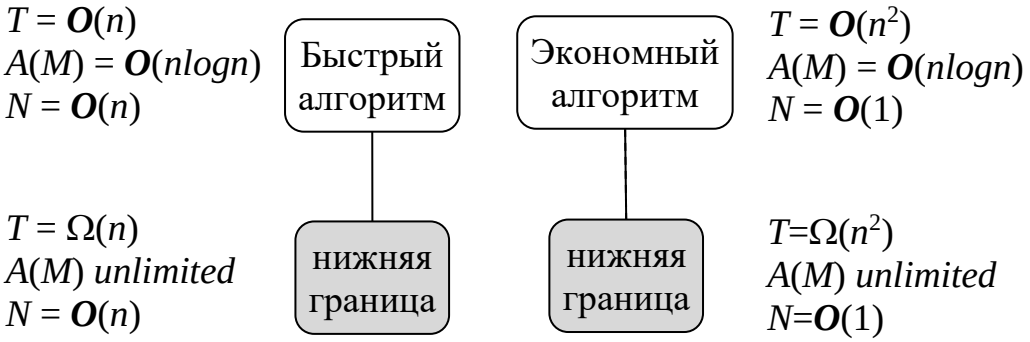


Рис. 1. Алгоритмы и оценки

Задача о кратчайшем пути является одной из важнейших классических задача теории графов. В случае взвешенного графа расстояние понимается как минимальная сумма весов дуг на пути между вершинами. Случай целых неотрицательных весов дуг легко сводится к рассматриваемому в данной статье

случаю не взвешенного графа, заменой каждой дуги на путь, длина которого равна весу пути. Эта задача имеет большое количество практических применений, в частности в протоколе динамической маршрутизации OSPF (*Open Shortest Path First*). Отображение, построению которого посвящена данная статья, полезно для решения многих задач, укажем две из них.

1) Симуляция распределенных алгоритмов для неориентированного графа. Для передачи сообщения из конца $b$ дуги $a \to b$ в ее начало $a$ нужно решить *проблему отката* (*backtracking problem*): посылать сообщение по ориентированному пути $b \to ... \to a$.

Ранее эта проблема изучалась в контексте обхода ориентированного графа роботом (конечным автоматом), двигающимся по дугам графа [2-4]. Эта модель эквивалентна «инвертированной» модели с одним сообщением, передаваемым по дугам, и идентичными роботами, находящимися в вершинах. Длина обхода графа без кратных ребер равна $\boldsymbol{O}(nm)$, где $m$ – число ребер, однако из-за проблемы отката в оценке алгоритмического обхода для разных алгоритмов добавляется слагаемое $\boldsymbol{O}(n^2 log n)$ [2], $\boldsymbol{O}(n^2 log log n)$ [3], а для наилучшего известного результата – $\boldsymbol{O}(n^2 log^* n)$ [4], где $log^*$ – итерированный логарифм.

Так можно симулировать любой алгоритм, разработанный для неориентированных систем, например [5-6]. Длина пути $b \to ... \to a$ не превосходит $n$-1, что увеличивает время работы не более чем в $n$-1 раз. Минимум достигается, если это кратчайший путь.

2) Самонастройка узлов программно-конфигурируемых сетей. Сетевой коммутатор (switch) хранит в программно-настраиваемых таблицах отображение адреса целевого хоста в номер выходного порта. Время передачи сообщений минимально, если в отображении использованы кратчайшие пути.

## 2. Определения и обозначения

*Маршрут* – последовательность дуг, в которой конец не последней дуги совпадает с началом следующей дуги. *Путь* – маршрут, в котором никакие две дуги не имеют общего конца, и конец последней дуги не равен началу первой дуги. *Расстояние* от вершины $a$ до вершины $b$ – длина кратчайшего пути от $a$ до $b$. *Прямое (обратное) дерево* – корневое дерево, дуги которого ориентированы от корня (к корню). *Прямое (обратное) дерево кратчайших путей* – подграф, являющийся прямым (обратным) деревом, в котором расстояние от корня до каждой вершины (от каждой вершины до корня) по дереву и по графу одинаковы. *Остовное дерево* – суграф, являющийся деревом.

Вершина описывается своим номером, а дуга – тройкой $(a,i,b)$, где $a$ – номер начала дуги, $i$ – номер дуги в вершине $a$, $b$ – номер конца дуги. Связный граф задается множеством троек, а маршрут – последовательностью троек, их дуг.

Обозначим: $r$ – корень графа, $d^-(a)$ и $d^+(a)$ – полустепени исхода и захода вершины $a$, $V(H)$ – множество начал и концов дуг из множества $H$.

### 3. Коррекция деревьев

Пусть все вершины графа $H$ достижимы из вершины $x$, и $D \subseteq H$ – прямое дерево с корнем в $x$. Определим коррекцию дерева, когда становится известной дуга $(a,i,b) \in H \setminus D$ с началом в $D$, т.е. $a \in V(D)$. Если $b \notin V(D)$, то дуга $(a,i,b)$ добавляется в $D$. Если $b \in V(D)$, то в $D$ есть единственная дуга $(a`,i`,b)$, заканчивающаяся в $b$. Если в дереве $D$ расстояние от $x$ до $a$, увеличенное на 1, меньше, чем расстояние от $x$ до $b$, то дуга $(a`,i`,b)$ заменяется дугой $(a,i,b)$. Очевидно, что если коррекция применяется последовательно для всех дуг графа, то будет получено прямое остовное (для $H$) дерево кратчайших путей с корнем в $x$.

Пусть из всех вершин графа $H$ достижима вершина $x$, и $D \subseteq H$ – обратное дерево с корнем в $x$. Определим коррекцию дерева, когда становится известной дуга $(a,i,b) \in H \setminus D$ с концом в $D$, т.е. $b \in V(D)$. Если $a \notin V(D)$, то дуга $(a,i,b)$ добавляется в $D$. Если $a \in V(D)$, то в $D$ есть единственная дуга $(a,i`,b`)$, начинающаяся в $a$. Если по дереву $D$ расстояние от $b$ до $x$, увеличенное на 1, меньше, чем расстояние от $a$ до $x$, то дуга $(a,i`,b`)$ заменяется дугой $(a,i,b)$. Очевидно, что если коррекция применяется последовательно для всех дуг графа, то будет получено обратное остовное (для $H$) дерево кратчайших путей с корнем в $x$.

### 4. Способы передачи сообщений

*Рассылка из корня*. Без параметров. Сообщение, начиная с корня, проходит по каждой дуге графа ровно один раз. Когда вершина $x \neq r$ получает сообщение первый раз, она посылает его по каждой исходящей из $x$ дуге. Повторные сообщение игнорируются.

*Множественная рассылка*. Параметры: номер вершины $x$, создавшей сообщение. Сообщение распространяется по графу аналогично рассылке из корня, но создателем сообщения может быть любая вершина, и таких вершин может быть несколько. Когда вершина $x$ первый раз получает сообщение, созданное вершиной $y \neq x$, она посылает сообщение по каждой исходящей из $x$ дуге. Вершина хранит (в виде битовой шкалы) множество $S(x)$ вершин-создателей, от которых получены сообщения.

*Рассылка по прямому дереву*. Параметры: описание прямого дерева $F(x)$ с корнем в вершине $x$. Сообщение передается по дугам дерева от $x$ до всех его вершин.

*Пересылка по прямому дереву*. Параметры: описание прямого дерева $F(x)$ с корнем в вершине $x$, номер вершины $y \in V(F(x))$. Сообщение передается по дугам дерева от $x$ до $y$.

*Пересылка по обратному дереву*. Параметры: описание обратного дерева $R(x)$ с корнем в вершине $x$. Сообщение передается по дугам дерева до $x$.

*Сбор по обратному дереву*. Параметры: описание обратного дерева $R(x)$ с корнем в вершине $x$, булевский признак $q$. Сообщение, полученное или

созданное вершиной $z \in V(R(x)) \setminus \{x\}$, запоминается в переменной $m(z)$, если там уже не запомнено сообщение с $q =$ ***true***. Когда вершина $z$ получит сообщение по каждой входящей в нее дуге дерева $R(x)$ и сама создаст сообщение, она посылает запомненное в $m(z)$ сообщение по исходящей дуге дерева $R(x)$. Среди полученных вершиной $x$ сообщений (включая созданное ею) будет сообщение с $q =$ ***true*** тогда и только тогда, когда хотя бы одна вершина из $V(R(x))$ создала сообщение с $q =$ ***true***.

### 4. Быстрый алгоритм сбора информации в корне

Сначала рассылкой из корня во все вершины передается сообщение *Старт* за время не более $n$ тактов. Это сообщение накапливает в себе описание пройденного им маршрута *Fp*. Когда вершина $x$ получает сообщение, она запоминает *Fp*. Множество запомненных в $x$ маршрутов образует подграф $H(x)$, состоящий из прямого дерева с корнем в $r$ и множества $In(x)$ дуг, ведущих из его листьев в $x$. Поэтому число дуг в $H(x)$ равно $\boldsymbol{O}(n)$, а размер описания $H(x)$ равно $\boldsymbol{O}(n \log n)$.

Получив *Старт*, вершина создает сообщение *Возврат*, которое пересылается по графу множественной рассылкой, что гарантирует его доставку в корень. На одной дуге может быть $\boldsymbol{O}(n)$ сообщений *Возврат*. *Возврат* содержит как параметр *Fp* из сообщения *Старт*, и тоже накапливает в себе описание пройденного им маршрута *Rp*.

Получая *Старт* с *Fp* (это цикл) или *Возврат* с *Fp* и *Rp* (вместе образуют цикл), корень формирует описание текущих прямого $F$ и обратного $R$ деревьев на множестве вершин $V = V(F) = V(R)$; размер описания равен $\boldsymbol{O}(n \log n)$. Для этого используется коррекция прямого дерева перебором дуг цикла *Fp* или *Fp·Rp* от начала к концу и коррекция обратного дерева перебором дуг цикла *Fp* или *Fp·Rp* от конца к началу.

Корень в цикле организует опрос вершин в старт-стопном режиме. Сообщение *Опрос* передается рассылкой по дереву $F$, а корень запоминает число $w := |V|$ вершин, которые получат *Опрос*. Получив *Опрос*, вершина $x$ создает сообщение *Ответ*, в котором указывает $H(x)$ и полустепень исхода $d^-(x)$. *Ответ* передается пересылкой по дереву $R$. Поэтому на одной дуге может быть $\boldsymbol{O}(n)$ сообщений *Ответ*.

Корень, получая $H(x)$, корректирует описания деревьев $F$ и $R$ (соответственно меняется $V$) с помощью коррекции прямого дерева перебором дуг из $H(x)$ от корня до $x$, и коррекции обратного дерева перебором дуг из $H(x)$ от $x$ до корня. Корень запоминает в двух массивах $D^+(x) := |In(x)|$ и $D^-(x) := d^-(x)$.

Когда корень получит *Ответ* от всех $w$ вершин, проверяется условие завершения алгоритма: 1) полустепень исхода $d^-(x)$ запомнена в $D^-(x)$ для каждой $x \in V$, 2) суммарное число известных входящих дуг равно суммарному числу исходящих дуг $\Sigma_{x \in V} D^+(x) = \Sigma_{x \in V} D^-(x)$. Если условие не выполнено, корень снова начинает опрос вершин, запоминая новое $w := |V|$.

Покажем, что алгоритм заканчивается через конечное время и определим его оценки. За время $t_1 \le n$ *Старт* пройдет по каждой дуге. После этого за время $t_2 \le n$-1 *Возврат* от каждой вершины дойдет до корня. Поэтому через время не более $t_1 + t_2$ будет $V = V(G)$. *Опрос* проходит дерево $F$, а *Ответ* – дерево $R$, за время не более $n$-1. Поэтому цикл *Опрос-Ответ* длится не более 2($n$-1) тактов, и через время не более $t_1 + t_2 + 2(n\text{-}1)$ начнется очередной цикл, в котором каждая вершина $x \ne r$ создает *Ответ* с $|In(x)| = d^+(x)$. После получения корнем всех *Ответов* на этот *Опрос* будет выполнено условие конца работы. Поэтому время работы алгоритма $T \le t_1+t_2+2(n\text{-}1)+2(n\text{-}1) = \boldsymbol{O}(n)$. Из описания алгоритма следует, что $N = \boldsymbol{O}(n)$, $A = \boldsymbol{O}(n\log n)$, $M = \boldsymbol{O}(n\log n)$.

Покажем, что в конце работы алгоритма $F$ и $R$ остовные деревья кратчайших путей. В этот момент времени $V = V(G)$, т.е. деревья $F$ и $R$ остовные. Кроме того, для каждой вершины $x \in V$ в корне хранится $D^-(x) = d^-(x)$. Поэтому условие окончания алгоритма может быть выполнено только в том случае, когда $D^+(x) = d^+(x)$ для каждой вершины $x \in V$. Следовательно, для каждой дуги $(y,i,x)$ корень получил такой *Ответ* от $x$, что $(y,i,x) \in H(x)$. В этот момент времени $x \in V$, а в $H(x)$ существует путь от корня, последняя дуга которого – это дуга $(y,i,x)$. Это значит, что каждая дуга графа участвует в коррекции деревьев $F$ и $R$. Следовательно, $F$ и $R$ деревья кратчайших путей.

### 5. Экономный алгоритм сбора информации в корне

Это модификация быстрого алгоритма, которая уменьшает оценку $N$ с $\boldsymbol{O}(n)$ до $\boldsymbol{O}(1)$ за счет увеличения времени работы не более чем в $n$ раз. Причиной оценки $N = \boldsymbol{O}(n)$ являются способы передачи сообщений *Возврат* – множественная рассылка, и *Ответ* – пересылка по дереву $R$. *Возврат* мы удалим. Теперь не будет времени $t_2$ и гарантии того, что корень узнает о всех вершинах графа через время $t_1 + t_2 = \boldsymbol{O}(n)$. Корень узнает о вершине только при получении *Старт* или *Ответ*. *Ответ* на данный *Опрос* будет передаваться не пересылкой по дереву $R$, а сбором по нему. При создании *Ответа* вершиной $x$ признак $q := (D^+(x) < |In(x)|)$ показывает, что множества $H(x)$ и $In(x)$ изменились по сравнению с тем, что известно корню. От каждой вершины $x$ может придти только один *Ответ* с истинным признаком $q$ и $|In(x)| = d^+(x)$. Поэтому после времени $t_1$ будет не более $n$ циклов *Опрос-Ответ*, после чего условие конца работы будет выполнено. Следовательно, время работы алгоритма $T \le t_1+2(n\text{-}1)+n(2(n\text{-}1)) = \boldsymbol{O}(n^2)$. Теперь на одной дуге может находиться либо не более одного сообщения *Старт*, либо не более одного сообщения *Опрос* и (если это дуга из $F \cap R$) не более одного сообщения *Ответ*. Поэтому $N \le 2 = \boldsymbol{O}(1)$.

### 6. Быстрый алгоритм построения отображения

Сначала применяем быстрый алгоритм сбора информации в корне: описаний прямого $F$ и обратного $R$ деревьев кратчайших путей. При этом в

каждой вершине $x$ создаётся описание $In(x)$ множества входящих дуг. Затем корень организует доставку в каждую вершину $x$ описания деревьев $F$ и $R$ с помощью сообщения *Остовы* рассылкой по дереву $F$. Каждая вершина $x$, получив *Остовы*, по $F$ и $R$ строит описание прямого $F(x)$ и обратного $R(x)$ остовных деревьев с корнем в $x$, и посылает в каждую вершину рассылкой по дереву $F(x)$ сообщение *Дуги* с описанием $F(x)$, $R$ и $In(x)$. Получая это сообщение от вершины $x$, вершина $y$ для каждой дуги из $In(x)$ корректирует деревья $F(y)$ и $R(y)$. Когда вершина $y$ получит *Остовы*, а также *Дуги* от всех вершин, $F(y)$ и $R(y)$ станут деревьями кратчайших путей с корнем в $y$, что позволит вершине $y$ создать требуемое отображение $f_y$. Тогда она посылает в корень сообщение *Конец* пересылкой по $R$. Корень определяет конец работы, когда получит *Конец* от всех вершин. Очевидно, этот этап работы не меняет оценки $T$, $A$, $M$ и $N$.

### 6. Экономный алгоритм построения отображения

Это модификация быстрого алгоритма, которая уменьшает оценку $N$ с $\boldsymbol{O}(n)$ до $\boldsymbol{O}(1)$ за счет увеличения времени работы не более чем в $n$ раз. Для этого, во-первых, вначале применяется не быстрый, а экономный алгоритм сбора информации в корне. Во-вторых, доставка информации из корня во все вершины выполняется не параллельно, а последовательно: корень организует цикл по вершинам. В этом цикле корень посылает *Остовы* одной вершине $x$ пересылкой по дереву $F$. *Конец* из вершины $y$ посылается корню сбором по дереву $R$ и означает теперь не завершение построения отображения в $y$, а только завершение обработки множества $In(x)$ в $y$. Алгоритм завершается, когда завершается цикл перебора вершин $x$. Поскольку этот цикл проходится $n$-1 раз, время увеличивается не более чем в $n$-1 раз. Поскольку в одном проходе цикла выполняется рассылка по дереву $F(x)$ только для одной вершины $x$, а *Конец* передается сбором по дереву $R$ (а не пересылкой), имеем $N = O(1)$. Оценки $A$ и $M$ не меняются.

### 6. Нижние оценки времени

Для быстрого алгоритма построения отображения оценка $T = \boldsymbol{O}(n)$ не может быть улучшена, даже если не ограничивать размер памяти вершины и сообщения, поскольку $T$ ограничено снизу временем распространения сообщения от одной вершины до другой, т.е. диаметром графа, который может достигать $n$-1.

Покажем, что для экономного алгоритма оценка $T = \boldsymbol{O}(n^2)$ также не улучшаема при любых размерах памяти вершин и сообщений.

Достаточно очевидно *утверждение 1*: если алгоритм на любом графе строит требуемое отображение в корне, то по каждой дуге графа должно пройти сообщение, после чего дойти до корня.

Оценка $N = \boldsymbol{O}(1)$ означает, что есть такая константа $k$, что на дуге одновременно находится не более $k$ сообщений. Вершина может послать

сообщение по дуге только, если на дуге гарантированно меньше $k$ сообщений. При произвольном времени прохода сообщения по дуге (но не более 1 такта) такую гарантию может дать только получение сообщения-подтверждения, прошедшего по дуге и далее по циклу вернувшегося в начало дуги. Отсюда *утверждение 2*: вершина, послав по дуге $p$ сообщений и получив $q$ подтверждений, посылает следующее сообщение по этой дуге при условии $p\text{-}q < k$.

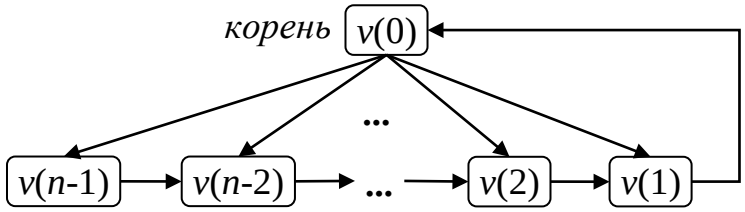


Рис. 2. Граф достижимости оценки $T = \Omega(n^2)$ для экономного алгоритма

Рассмотрим граф на рис. 2. Для $i > 0$ сообщение назовем $\{i\}$-сообщением, если $v(i)$ – самая левая вершина, через которую прошло сообщение. Скажем, что в вершине $v(i)$ происходит *склейка* $\{j`\}$-сообщения с $\{j\}$-сообщением, где $j` > j$, если $j = i$ и первое сообщение, которое посылает вершина $v(i)$, это $\{j`\}$-сообщение: оно «склеено» с $\{i\}$-сообщением, или $j > i$ и вершина, еще не послав $\{j\}$-сообщение, посылает $\{j`\}$-сообщение: оно «склеено» с $\{j\}$-сообщением. *Утверждение 3*: если алгоритм удовлетворяет утв. 1 и 2, то можно подобрать нумерацию дуг и вершин на рис. 2, чтобы в любой вершине $v(i)$, где $i > 0$, не было склейки сообщений до получения вершиной первого подтверждения. Доказательство можно посмотреть в [7].

Докажем *итоговое утверждение*: время работы алгоритма на графе на рис. 2 равно $\Omega(n^2)$. По утв. 3 считаем, что в вершине $v(i)$, где $i > 0$, нет склейки сообщений до получения этой вершиной первого подтверждения. Корень получит $\{1\}$-сообщение через $t_1 \geq 2$ такта. Обозначим $x_1 = 1$. Пусть вершина $v(x_i)$ посылает $1 \leq y_j \leq k$ сообщений до того, как она перестает посылать сообщения и начинает ждать первого подтверждения. Обозначим $x_{j+1} = x_j + y_j$, тогда $x_{j+1} \leq x_j + k$. Если $x_{j+1} \leq n\text{-}1$, то, поскольку склеек нет, $\{x_{j+1}\}$-сообщение не посылается из вершины $v(x_j)$ до получения ею первого подтверждения. Обозначим через $t_j$ время от начала работы, через которое корень получает $\{x_j\}$-сообщение. Только после этого не менее чем через 1 такт вершина $v(x_j)$ получит подтверждение, и только после этого $\{x_{j+1}\}$-сообщение может пройти путь до корня длиной не менее $x_j$. Следовательно, $t_{j+1} \geq t_j + 1 + x_j$. Имеем:

$x_1 = 1$, $t_1 \geq 2$,

$x_1 + 1 \leq x_2 \leq x_1 + k$, $t_2 \geq t_1 + 1 + x_1$,

...

$x_{u-1}+1 \le x_u \le x_{u-1}+k, \qquad t_u \ge t_{u-1} + 1 + x_{u-1}$

Тогда $i \le x_i$ и $x_u \le 1+k(u-1)$. Пусть $n \ge k + 1$. Выберем $u$ максимальное так, чтобы оставаться в диапазоне индексов: $0 \le n-1-k < x_u \le n-1$. Тогда $n-1-k < 1+k(u-1)$, что влечет $u > (n-2)/k$. Очевидно, $t(n) \ge t_u$. Имеем:

$$t(n) \ge 2 + (1+x_1) + (1+x_2) + (1+x_3) + ... + (1+x_u) = 2 + u + (x_1+x_2+x_3+...+x_u) \ge$$

$$\ge 2 + u + (1+2+3+...+u) = 2+u(u+3)/2 >$$

$$> 2+((n-2)/k)(((n-2)/k)+3)/2 = (n-2)^2/(2k^2)+3(n-2)/2k+2 \ge n^2/(3k^2) = \Omega(n^2).$$

Утверждение доказано.

Работа выполнена при поддержке Российского фонда фундаментальных исследований, проекты 17-07-00682-а и 16-07-01106-а.

**Литература**

1. И. Бурдонов, А. Косачев. Размер памяти для хранения упорядоченного корневого графа // Труды Института системного программирования РАН. Том 29, выпуск 2, 2017, стр. 7-26.
2. Y. Afek and E. Gafni. Distributed Algorithms for Unidirectional Networks // SIAM J. Comput., Vol. 23, No. 6, 1994, pp. 1152-1178.
3. И.Б.Бурдонов. Обход неизвестного ориентированного графа конечным роботом // «Программирование», 2004, №4, стр.11-34.
4. И.Б.Бурдонов. Проблема отката по дереву при обходе неизвестного ориентированного графа конечным роботом // «Программирование», 2004, №6, стр.6-29.
5. И. Бурдонов, А. Косачев. Общий подход к решению задач на графах коллективом автоматов // Труды Института системного программирования РАН. Том 29, выпуск 2, 2017, стр. 27-76.
6. И. Бурдонов, А. Косачев. Распределённые алгоритмы на корневых неориентированных графах // Труды Института системного программирования РАН. Том 29, выпуск 5, 2017, стр. 283-310.
7. И. Бурдонов, А. Косачев. Проблема отката в ориентированной распределенной системе // Труды Института системного программирования РАН. Том 30, выпуск 2, 2018, стр. 167-194.

**References**

1. I. Burdonov, A. Kossatchev. Razmer pamiati dlia khraneniia uporiadochennogo kornevogo grafa // Trudy Instituta sistemnogo programmirovaniia RAN. Tom 29, vypusk 2, 2017, str. 7-26.
2. Y. Afek and E. Gafni, Distributed Algorithms for Unidirectional Networks, SIAM J. Comput., Vol. 23, No. 6, 1994, pp. 1152-1178.
3. I.B.Bourdonov. Traversal of an Unknown Directed Graph by a Finite Robot. Programming and Computer Software,Vol. 30, No. 4, 2004, pp. 188-203.

4. I.B.Bourdonov. Backtracking Problem in the Traversal of an Unknown Directed Graph by a Finite Robot. Programming and Computer Software,Vol. 30, No. 4, 2004, pp. 305-322.
5. I. Burdonov, A. Kossatchev. Obshchii podkhod k resheniiu zadach na grafakh kollektivom avtomatov // Trudy Instituta sistemnogo programmirovaniia RAN. Tom 29, vypusk 2, 2017, str. 27-76.
6. I. Burdonov, A. Kossatchev. Raspredelennye algoritmy na kornevykh neorientirovannykh grafakh // Trudy Instituta sistemnogo programmirovaniia RAN. Tom 29, vypusk 5, 2017, str. 283-310.
7. I. Burdonov, A. Kossatchev. Problema otkata v orientirovannoi raspredelennoi sisteme // Trudy Instituta sistemnogo programmirovaniia RAN. Tom 30, vypusk 2, 2018, str. 167-194.