ЛИТЕРАТУРА

1. *Е. А. Диниц, М. А. Кронрод.* Один алгоритм решения задачи о назначении.— «Доклады АН СССР», 1969, т. 189, № 1.
2. *Г. Кун.* Венгерский метод решения задачи о назначении.— «Методы и алгоритмы решения транспортной задачи». Госстатиздат, 1963.
3. *I. Edmonds, R. M. Karp.* Theoretical improvements in algorithmic efficiency for network flow problems.— «J. of ACM», 1972, v. 19, N 2.
4. *В. П. Гришухин.* Об одном классе алгоритмов сравнения.— «Исследования по дискретной математике». М., «Наука», 1973.
5. *С. М. Бородкин.* О решении минимаксной задачи о назначении.— «Автоматика и телемеханика», 1974, № 10.

А. В. Карзанов

СПРАВОЧНАЯ ДЛЯ ВЫБОРКИ МАКСИМАЛЬНОГО ЭЛЕМЕНТА И ЕЕ ПРИЛОЖЕНИЯ

1. При алгоритмическом решении ряда задач дискретной математики возникает потребность в решении следующей задачи **А**.

Имеется совокупность объектов, каждому из которых отнесено число. Производится многошаговый процесс. На каждом шаге требуется выбрать объект с максимальным (минимальным) числом. На шаге из совокупности удаляются некоторые объекты и добавляются новые (не бывшие ранее).

Пусть N — число объектов, побывавших в совокупности за рассматриваемое число шагов. В [1] описан способ организации информации, применяя который можно решить задачу **А** с оценкой числа действий $O(N\cdot\log_2 N)$.

Предлагается другой алгоритм решения задачи **А** с той же оценкой — метод неоднородных справочных. Справочная [1] рассчитана на универсальное применение. Поэтому в каждый момент совокупность имеющихся объектов поддерживается в ней в упорядоченном виде, что довольно «дорого стоит». Так как в задаче **А** от справочной требуется только выдавать максимальный элемент, то в неоднородной справочной совокупность поддерживается в упорядоченном виде лишь частично, а доупорядочение производится только для объектов, близких к максимальному, по мере необходимости. Благодаря этому неоднородная справочная экономнее справочной [1].

Далее в работе даны два приложения неоднородной справочной. В п. 3 описан алгоритм нахождения окружности минимального диаметра, заключающей n заданных точек плоскости. Оценка алгоритма — $O(n\cdot\log_2 n)$. Ранее был известен алгоритм с оценкой $O(n^2)$ [2]. В предлагаемом алгоритме решается задача **А**. В п. 4 указано, как применить метод неоднородных справочных к алгоритму Дийкстры поиска кратчайших путей в графе [3] с тем, чтобы оценка числа действий стала $O(np\cdot\log_2 n)$ вместо $O(n^3)$, где n — число вершин, p — число дуг графа. Такая модификация выгодна при относительно невысокой «густоте» дуг графа.

2. Каждый объект будем идентифицировать парой $r = (A(r), x(r))$, где $A(r)$ — адрес объекта, $x(r)$ — число, по которому ведется сравнение. Считаем, что в информацию об объекте включен текущий адрес пары.

Устройство неоднородной справочной. Основная часть справочной — массив M. Элементами массива являются, во-первых, все пары, имеющиеся в совокупности; во-вторых, некоторые пары объектов, уже удаленных из совокупности (в последнем случае адреса объектов пусты). Пары в массиве расположены по убыванию чисел. Для части объектов в совокупности их пары не содержатся в M, в этом случае они объединены в семейства. Каждое семейство S_r отнесено к некоторой паре $r = (A(r), x(r))$ из M, разные семейства относятся к разным парам. Для каждой пары $\tilde{r} \in S_r$ должно выполняться $x(\tilde{r}) \geqslant x(r)$, а также $x(\tilde{r}) \leqslant x(r')$, где r' — пара, находящаяся в M непосредственно перед r (если r — не первая пара в M).

Каждое семейство задано ссылочным списком (определение списка дано в работе «Экономные реализации алгоритмов Эдмондса нахождения паросочетания максимальной мощности и максимального веса» из настоящего сборника).

В начальной справочной все пары содержатся в M (т. е. все семейства пусты).

Описание шага. Просматриваем первую пару (r) в M.

Случай I. Семейство пары r пусто (т. е. $x(r)$ — максимальное число по всем парам справочной). Если адрес $A(r)$ пуст, то выбрасываем r из M и шаг считаем законченным. Если адрес $A(r)$ непуст (т. е. соответствующий объект содержится в совокупности), то:

1) последовательно просматриваем пары объектов, подлежащих удалению;

а) если пара находится в M, то стираем у нее адрес объекта, а число оставляем;

б) если пара находится в семействе, то удаляем ее оттуда;

2) Для каждого возникающего объекта образуем соответствующую ему пару. Для каждой из этих пар $(\tilde{r})$ находим семейство $S_{r'}$ (быть может, до этого пустое), в которое его следует поместить (для пар r', $\tilde{r}$ и предыдущей к r' в M пары r'' должно выполняться: $x(\tilde{r}) \geqslant x(r')$ и $x(\tilde{r}) \leqslant x(r'')$). Для этого используем стандартную процедуру поиска в упорядоченном массиве числа, ближайшего и не превосходящего заданное (метод деления пополам): сравниваем $x(\tilde{r})$ с числом «средней» пары из массива, затем в зависимости от результата сравниваем $x(\tilde{r})$ со «средним» элементом «верхнего» либо «нижнего» подмассива и т. д. Число действий на работу с парой $\tilde{r}$ — $O(\log_2 |M|)$.

Случай II. Семейство S_r непусто. Упорядочиваем пары из S_r по убыванию чисел и наращиваем массив M

«сверху» парами семейства S_r в соответствии с этим упорядочением. Затем идем в новое начало массива и поступаем, как описано в I.

Оценка трудоемкости алгоритма. Пусть за рассматриваемое число шагов в совокупности побывало N объектов. Образование и удаление самих объектов из совокупности относим к внешней задаче и действия на них не учитываем.

1). На образование пар тратится $O(N)$ действий.

2). На образование начальной справочной достаточно $O(N_0 \cdot \log_2 N_0)$ действий, где N_0 — число объектов в начальной совокупности.

3). На занесение нового объекта в семейство тратится, как указывалось, $O(\log_2 |M|)$ действий, и поскольку $|M| < N$, то общее число таких действий за рассматриваемое число шагов — $O(N \cdot \log_2 N)$.

4). Пусть $S_{r_1}, S_{r_2}, \ldots, S_{r_l}$ — семейства, подвергавшиеся упорядочению (процедура II) за рассматриваемое число шагов. По условию эти семейства не содержат общих элементов, откуда $\sum\limits_{i=1}^{l} S_{r_i} < N$. Число действий на упорядочение семейства S_{r_i} — $O(|S_{r_i}| \cdot \log_2 |S_{r_i}|)$, откуда общее число действий по процедуре II — $O\left(\sum\limits_{i=1}^{l} |S_{r_i}| \cdot \log_2 |S_{r_i}|\right)$, или $O(N \cdot \log_2 N)$.

Все не разобранные нами группы действий имеют порядок трудоемкости ниже $N \cdot \log_2 N$. Таким образом, оценка трудоемкости алгоритма — $O(N \cdot \log_2 N)$.

3. Рассмотрим задачу нахождения окружности $\bar{S}$ минимального диаметра, заключающей n заданных точек плоскости.

Из элементарной геометрии известно, что в данном множестве M (из n точек) либо найдутся 3 точки, лежащие на $\bar{S}$ и образующие остроугольный треугольник, либо 2 точки, диаметрально противоположные на $\bar{S}$.

Алгоритм состоит из двух частей:

1. Выделение выпуклой линейной оболочки (в.л.о.) множества M. Здесь будет показано также, что в общем случае этого нельзя сделать меньше чем за $C \cdot n \cdot \log_2 n$ действий ($C > 0$).

2. Построение окружности минимального диаметра для в.л.о. Очевидно, $\bar{S}$ для M и для в.л.о. совпадают.

Выделение выпуклой линейной оболочки. В. л. о. множества M будем искать в виде перечня вершин в.л.о., упорядоченных против часовой стрелки.

Будем считать, что точки заданы в декартовой системе координат. Выберем точки a, $b \in M$ соответственно с наибольшей и наименьшей абсциссами и найдем точку O — середину отрезка $[a, b]$. Очевидно, точки a и b лежат на в.л.о., а O — внутри (в вырожденном случае — тоже на в.л.о.).

Вычислим углы радиус-векторов $\overrightarrow{Om}$, где m пробегает множество M, с осью абсцисс (углы будем измерять в интервале от 0 до 2π).

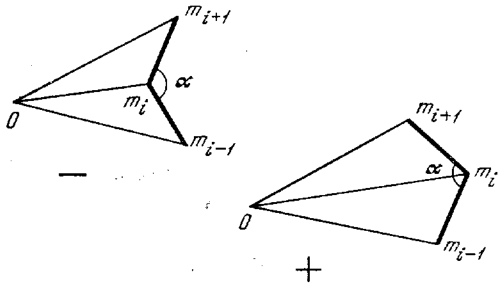

Рис. 1

Упорядочим эти углы по возрастанию ($O(n \cdot \log_2 n)$ действий). Если какие-то точки будут иметь равные углы, то, очевидно, в в.л.о. может быть только одна точка с радиус-вектором наибольшей длины, так что остальные точки можно попросту отбросить.

В результате будем иметь последовательность K точек $m_1, m_2, \ldots, m_{n'}$ ($n' \leqslant n$), упорядоченную против часовой стрелки.

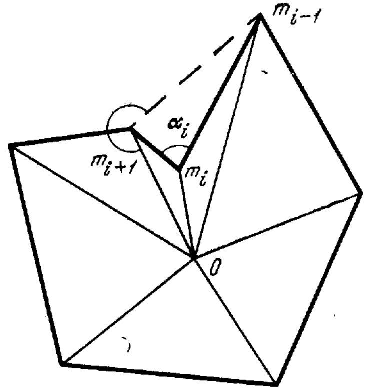

Рис. 2

Для каждой тройки $\{m_{i-1}, m_i, m_{i+1}\}$ соседних в K точек (считаем, что m_1 и $m_{n'}$ тоже соседние) вычислим угол $\alpha_i = \angle\, m_{i-1}, m_i m_{i+1}$ ($|\alpha| \leqslant \pi$). Углу α_i припишем знак $+$, если т. O лежит «внутри» α_i, и $-$ в противоположном случае (рис. 1). Если $\alpha_i = \pi$, то можем приписывать ему знак $+$ или $-$ в зависимости от того, хотим мы оставлять в в.л.о. вершины, образующие угол π, или нет (для нахождения окружности минимального диаметра, очевидно, не имеет смысла оставлять такие точки). Знак можно вычислить, подставляя в левые части уравнения прямой $m_{i-1}m_{i+1}$ координаты точек O и m_i. Если знаки совпадут, то α_i — отрицателен. Легко видеть, что если все углы положительны, то K является в.л.о. Также очевидно, что если α_i отрицателен, то m_i не войдет в в.л.о.

Выпишем в произвольной последовательности все отрицательные углы. Пусть первым идет угол α_i (рис. 2). Выкинем из K точку m_i и пересчитаем углы α_{i-1} и α_{i+1}. Знаки этих углов могут измениться (но только с $+$ на $-$). Если появились новые отрицательные углы (один или два), запишем их в конец последовательности и перейдем к следующему углу. Число таких шагов не будет превышать $n' - 2$, так как каждому будет соответствовать выкидывание точки из K. Каждый шаг требует $O(1)$ действий. Когда отрицательные углы исчезнут, оставшиеся в K точ-

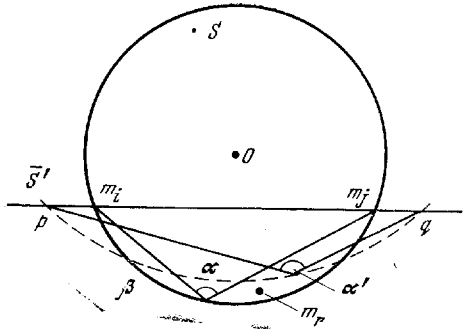

Рис. 3

ки определят в.л.о. Итак, общее число действий на выделение в.л.о.— $O(n\cdot\log_2 n)$.

Покажем, что число действий любого алгоритма нахождения в.л.о. не меньше чем $C\cdot n\cdot\log_2 n$.

Возьмем произвольный набор положительных чисел $a_1, a_2, \ldots, a_n$. Пусть a_k — максимальное число. Образуем числа $\beta_i = \frac{a_i}{a_k}\cdot 2\pi$. Каждому числу a_i отнесем точку C_i на плоскости, удаленную от начала координат на расстояние 1, такую, что угол $\overline{Oc_i}$ с осью абсцисс равен β_i. Очевидно, все точки принадлежат в.л.о. В то же время, когда в.л.о. будет найдена, β_i (а с ними и a_i) окажутся упорядочены по возрастанию. Известно же, что упорядочить n чисел нельзя меньше чем за $C\cdot n\cdot\log_2 n$ действий (C — некоторая константа).

Построение окружности минимального диаметра, заключающей выпуклую линейную оболочку. Докажем сначала следующие леммы.

Лемма 1. Пусть существует окружность S, заключающая множество M и проходящая через m_i и m_j, и пусть β — та из 2 дуг окружности, вписанный в которую угол $\alpha \geqslant \frac{\pi}{2}$ (рис. 3). Искомая окружность $\bar{S}$ не изменится при выкидывании из M множества точек, заключенных в секторе $m_i\beta m_jO$ (кроме, разумеется, m_i и m_j).

Доказательство. Предположим, при выкидывании точек из сектора $m_i\beta m_jO$ окружность минимального диаметра стала другой ($\bar{S}'$), т. е. найдется точка m_r в секторе, которая не будет заключена в $\bar{S}'$. Пусть $\bar{S}'$ пересекает прямую m_im_j в точках p и q.

Между радиусом окружности (R), длиной хорды (l) и вписанным углом (γ), опирающимся на хорду, имеется соотношение:

$$R = \frac{l}{2\sin\gamma}. \tag{1}$$

В нашем случае $R_S = \frac{|m_im_j|}{2\sin\alpha}$, $R_{\bar{S}'} = \frac{|pq|}{2\sin\alpha'}$, где α' — угол, вписанный в дугу pq окружности $\bar{S}'$.

Очевидно, $|m_im_j| \leqslant |pq|$, легко видеть также, что $\pi > \alpha' > \angle pm_rq \geqslant \angle m_im_rm_j \geqslant \alpha \geqslant \frac{\pi}{2}$. Отсюда $R_S < R_{\bar{S}'}$, что противоречит $R_S \geqslant R_{\bar{S}} \geqslant R_{\bar{S}'}$. ■

Лемма 2. Рассмотрим все тройки соседних точек в в.л.о. Если $\{m_{l-1}, m_l, m_{l+1}\}$ такова, что окружность S_l, проходящая через точки m_{l-1}, m_l, m_{l+1}, имеет максимальный радиус по всем тройкам, то вся в.л.о. заключена в S_l.

Доказательство. Пусть m' — точка, диаметрально противоположная m_l, и β — одна из 2 дуг, образуемых точками m_l и m'. Предположим, существуют точки из в.л.о., находящиеся за пределами β. Пусть m_k — первая такая вершина (в направлении от m_l). Отрезок $[m_l, m_k]$ пересекается с S_l в точке t (рис. 4). Рассмотрим все тройки, начиная с $\{m_l, m_{l+1}, m_{l+2}\}$ и кончая $\{m_{k-2}, m_{k-1}, m_k\}$. Пусть $\{m_{l'-1}, m_{l'}, m_{l'+1}\}$ — тройка с максимальным радиусом описанной окружности ($S_{l'}$) среди рассматриваемых.

Покажем, что внутри $S_{l'}$ не могут содержаться все вершины промежутка $m_l \div m_k$. Обозначим угол, вписанный в дугу $\widehat{m_lt}$, через α. Точки пересечения $S_{l'}$ с прямой m_lm_k обозначим через p и q. Если $S_{l'}$ содержит все вершины от m_l до m_k, то $|pq| \geqslant |m_lm_k| > |m_lt|$. Кроме того, $\pi > \angle pm_{l'}q > \angle m_lm_{l'}t \geqslant \alpha > \frac{\pi}{2}$, и, используя формулу (1) из предыдущей леммы, получим $R_{S_{l'}} > R_{S_l}$, что невозможно. Следовательно, часть в.л.о. от $m_{l'}$ до m_k или часть от $m_{l'}$ до m_l не содержится целиком в $S_{l'}$.

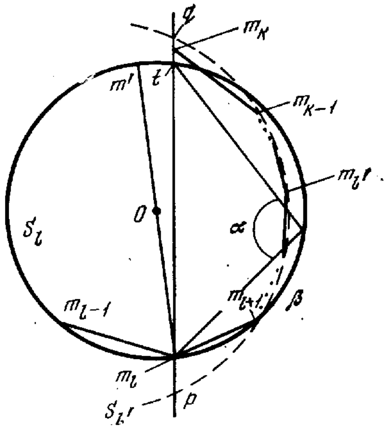

Рис. 4

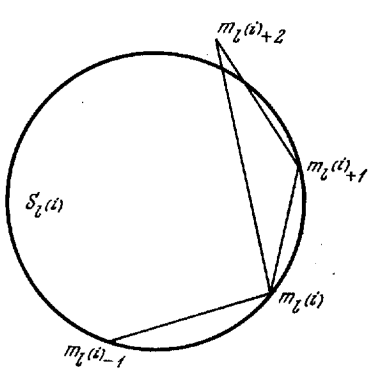

Рис. 5

Допустим, это $m_{l'} \div m_k$. Очевидно, в этой части число троек меньше, чем в $m_l \div m_k$. Применим аналогичные рассуждения к $m_{l'} \div m_k$ и окружности $S_{l'}$ и покажем, что описанная окружность максимального радиуса среди этих троек не содержит целиком часть в.л.о. от $m_{l'}$ до m_k и т. д. В конце концов дойдем до случая, когда окружность $S_{l(i)}$, имеющая максимальный радиус для троек в части в.л.о. $m_{l(i-1)} \div m_{k(i-1)}$, не содержит $m_{l(i)+2}$ (или $m_{l(i)-2}$) (рис. 5).

Здесь применение формулы (1) дает $R_{S_{l(i)}} < R_{S(\Delta m_{l(i)} m_{l(i)+1} m_{l(i)+2})}$, что противоречит выбору $S_{l(i)}$. ■

Изложим теперь принцип алгоритма нахождения $\overline{S}$ для в.л.о. Вначале вычислим радиусы описанных окружностей всех троек (на это требуется $O(n)$ действий). Выберем тройку с максимальным радиусом (пусть это $\{m_{l-1} m_l m_{l+1}\}$).

Окружность S_l по лемме 2 будет заключать всю в.л.о. Если треугольник $m_{l-1} m_l m_{l+1}$ — остроугольный или прямоугольный, то $\overline{S} = S_l$. Если $\angle m_l m_{l-1} m_{l+1}$ (или $\angle m_l m_{l+1} m_{l-1}$) тупой, то, как нетрудно видеть, $\overline{S}$ имеет

своим диаметром отрезок $[m_l, m_{l-1}]$ (соответственно $[m_l, m_{l+1}]$). Если же $\angle m_{l-1} m_l m_{l+1} > \frac{\pi}{2}$, то по лемме 1 точку m_l можно выкинуть из в.л.о. При этом распадутся тройки $\{m_{l-1} m_l m_{l+1}\}$, $\{m_{l-2} m_{l-1} m_l\}$ и $\{m_l m_{l+1} m_{l+2}\}$ и образуются 2 новые $\{m_{l-2} m_{l-1} m_{l+1}\}$ и $\{m_{l-1} m_{l+1} m_{l+2}\}$. Подсчитываем радиусы описанных окружностей для этих троек. Следующим шагом снова находим тройку с максимальным радиусом и либо найдем искомую окружность $\overline{S}$, либо выкинем еще одну вершину из в.л.о. и т. д. Таких шагов будет не больше n.

Таким образом, приходим к задаче **А**, где объектами будут тройки, а сравнивающимися числами — радиусы описанных окружностей.

На каждом шаге возникают две новые тройки, следовательно, число объектов, побывавших в совокупности за всю работу, не превышает $3n$. Таким образом, оценка трудоемкости алгоритма нахождения окружности минимального диаметра с учетом оценки решения задачи **А** — $O(n \cdot \log_2 n)$.

4. Имеется связный ориентированный граф $G = [V; U]$ без петель и кратных дуг с множеством вершин V, $|V| = n$ и множеством дуг U, $|U| = p$. Для каждой дуги u задана ее длина $a(u) \geqslant 0$. Требуется для каждой упорядоченной пары вершин x, y найти кратчайший по длине путь $L_{x,y}$ с началом в x и концом в y (длиной $l(L)$ пути L называется $\sum\limits_{u \in L} a(u)$).

Дадим краткое описание алгоритма Дийкстры [3]. Этапом в алгоритме будем называть процесс нахождения кратчайших путей от фиксированной вершины v до всех остальных. Этап состоит из итераций. Пусть несколько итераций уже проведено. Перед началом следующей итерации имеются:

а) множество P просмотренных вершин. Для каждой просмотренной вершины x кратчайший путь $L_{v,x}$ уже найден;

б) множество Q помеченных вершин. Каждая помеченная вершина y соединена с какой-либо просмотренной вершиной входящей дугой, и для y известен путь $L(y)$ минимальной длины среди путей вида $L_{v,x}$, (x, y), y, где x — просмотренная вершина, а также число $l(y)$ — длина пути $L(y)$. Такую дугу (x, y) назовем *направляющей* для y.

Вначале полагаем $P = \phi$, $Q = \{v\}$. На очередной итерации:

1) среди помеченных вершин отыскивается вершина y, для которой $l(y)$ минимально; y удаляется из Q и присоединяется к P (доказывается, что $L(y)$ — кратчайший путь);

2) перебираются исходящие из y дуги, не ведущие в уже просмотренные вершины. Пусть (y, z) — очередная дуга;

а) если $z \notin Q$, то присоединяем z к Q и полагаем $L(z) = L(y), (y, z), z$ и $l(z) = l(y) + a(y, z)$;

б) если $z \in Q$, то определяем величину $l_y(z) = l(y) + a(y, z)$. Если $l(z) \leqslant l_y(z)$, то $L(z)$ и $l(z)$ оставляем прежними; если $l(z) > l_y(z)$, то полагаем $l(z) := l_y(z)$ и $L(z) := L(y), (y, z), z$.

Этап оканчивается, как только все вершины окажутся просмотренными.

Число действий группы 2) за итерацию пропорционально числу исходящих из y дуг, следовательно, число действий за весь этап — $O(p)$. Поиск вершины y в Q с минимальным $l(y)$ требует $O(|Q|)$ или $O(n)$ действий, следовательно, число действий группы 1) за этап — $O(n^2)$.

Началу каждой итерации отнесем совокупность, объектами которой являются пары: вершина из Q и ее направляющая дуга. Объекту $\{y, (x, y)\}$ соответствует число $l(y)$. Таким образом, работа алгоритма в части 1) на этапе может рассматриваться как решение задачи **А** для указанной совокупности, для чего можно применить метод неоднородных справочных. Число объектов, побывавших

в совокупности, за этап равно числу различных направляющих дуг, откуда оценка числа действий группы 1) за этап, а следовательно, и всего этапа — $O(p \cdot \log_2 p)$, или $O(p \cdot \log_2 n)$.

ЛИТЕРАТУРА

1. *Г. М. Адельсон-Вельский, Е. М. Ландис.* Один алгоритм организации информации.— «Доклады АН СССР», 1962, т. 146, № 2.
2. *Р. С. Гутер, Д. В. Полунов, И. А. Фараджев.* Алгоритм решения одного класса задач нелинейного программирования.— «Теория оптимальных решений», вып. 4, Киев, 1969.
3. *E. W. Dijkstra.* A note on two problems in connection with graphs.— «Numerical Mathematik», 1959, v. 1.

М. Ф. Филлер

ОДИН АЛГОРИТМ НАХОЖДЕНИЯ КРАТЧАЙШИХ ПУТЕЙ

Пусть задан ориентированный граф [1] G с множеством вершин V, $|V| = n$, множеством дуг U, $|U| = p$, и с неотрицательной функцией $l(u)$ длин дуг. В графе G выделена начальная вершина s, и нас интересуют кратчайшие пути из s в остальные вершины $v \in V \setminus s$ и расстояния $r(v)$ до этих вершин (т. е. длины кратчайших путей) [2].

Опишем алгоритм, решающий поставленную задачу с трудоемкостью $O(p)$ действий [3] для важного класса графов, в которых длины дуг не сильно различаются между собой. Более точно, в общем случае трудоемкость

---

[1] Для неориентированного графа все аналогично.
[2] Обзор алгоритмов нахождения кратчайших путей см. в [2].
[3] Имеются в виду действия абстрактной ЭВМ.