УДК 519.85(023)+372.8:51

ЗАДАЧА СОРТИРОВКИ НА ГРАФАХ В ОЛИМПИАДАХ ПО ПРОГРАММИРОВАНИЮ

М.И. Киндер[1], А.В. Казанцев[2]

Казанский (Приволжский) федеральный университет, г. Казань

[1] mkinder@rambler.ru, [2] Andrei.Kazantsev@kpfu.ru

Аннотация

Разобрана задача сортировки данных, отношение порядка между которыми описано в виде отношения смежности вершин на произвольном графе. Выделены подзадачи и вопросы, относящиеся к «окрестности» проблемы; их решение представляет собой своеобразные уровни «погружения» в решение общей задачи. Обсуждены алгоритмы решения отдельных подзадач для графов специального вида, а также различные подходы к решению проблемы сортировки в общем случае. Задача сортировки такого типа предлагалась на Кубке международной школы ISI-Junior по спортивному программированию в июле 2019 года (г. Иннополис).

Ключевые слова*: олимпиады по информатике, олимпиады по математике, олимпиады по спортивному программированию, многоуровневые задачи, исследовательские задачи для школьников, задача сортировки на графах*

ВВЕДЕНИЕ

Сортировка является одной из фундаментальных алгоритмических проблем в информатике. В настоящее время имеется огромный выбор различных алгоритмов сортировки, в которых используются многие важные методы. Как правило, в задачах на сортировку требуется по заданной последовательности чисел (объектов) построить новую последовательность, в которой числа расположены в порядке возрастания или убывания. В такой формулировке считается, что новая последовательность элементов подчиняется некоторому исходному отношения порядку.

В этой статье мы разбираем задачу сортировки данных в случае, когда отношение порядка задано в виде отношения смежности вершин на произвольном графе.

1. ПОСТАНОВКА ЗАДАЧИ

На одном из соревнований по программированию (Международная школа ISI-Junior) школьникам была предложена следующая задача (автор — *Киндер М.И.*).

В IT-компании работают n сотрудников, пронумерованных от 1 *до n. Некоторые сотрудники этой компании находятся в подчинении у других сотрудников. У одного и того же сотрудника может быть несколько начальников, и никакой сотрудник не может быть начальником самому себе.*

Эффективность работы каждого сотрудника оценивается с помощью специального показателя его рейтинга.

У каждого сотрудника есть свой рабочий кабинет, номер которого в настоящий момент совпадает с номером этого сотрудника. Для улучшения работы компании решено провести реформу, упорядочив номера кабинетов в соответствии с рейтингом сотрудников, при этом сотрудник с самым низким рейтингом после реформы должен оказаться в кабинете 1*, сотрудник со вторым по величине рейтингом – в кабинете* 2*, и так далее, наконец, сотрудник с самым высоким рейтингом должен переселиться в кабинет n.*

Корпоративные правила компании разрешают пересаживать сотрудников A и B только в том случае, если один из них находится в подчинении у другого (то есть либо A находится в подчинении сотрудника B, либо B находится в подчинении у сотрудника A).

Необходимо составить программу, которая определяет, возможна ли требуемая рассадка персонала и, если возможна, определить список этих пересаживаний.

2. МАТЕМАТИЧЕСКАЯ МОДЕЛЬ ЗАДАЧИ

Рассмотрим граф G, в котором вершины обозначают сотрудников компании, а ребра соответствуют отношению подчинения, то есть вершины A и B соединены ребром, если сотрудник A находится в подчинении сотрудника B, либо B находится в подчинении у сотрудника A. Кроме того,

- каждая вершина графа имеет рейтинг — целое положительное число;
- разрешается менять местами рейтинги у двух смежных вершин графа;
- требуется отсортировать исходный набор рейтингов по возрастанию, переставляя на каждом шаге рейтинги только у двух вершин; если это сделать невозможно, то вывести число –1.

Замечание. Обычная сортировка соответствует случаю, когда граф отношений между элементами исходного массива является полным, то есть любые две вершины графа соединены ребром. Действительно, в этом случае любые два числа из массива рейтингов можно переставить местами без всяких ограничений.

3. АЛГОРИТМЫ РЕШЕНИЯ ЗАДАЧИ

Обсудим возможные подходы к решению задачи в частных случаях, а затем разберём алгоритм решения в общей ситуации. Прежде всего, отметим, что для сортировки всего массива рейтингов достаточно разобраться в процедуре перестановки каких-нибудь двух чисел этого массива.

Подзадача 1. Рассмотрим случай, когда в графе *G* отношений есть вершина, соединённая ребрами со всеми вершинами графа. Такую вершину будем называть *боссом* и обозначим её через *B*. Эта вершина имеет статус *начальника* по отношению ко всем без исключения сотрудникам компании.

В этом случае перестановку любых двух сотрудников с рейтингами *x* и *y* можно сделать за один или три шага, при этом положение остальных элементов массива рейтингов не меняется:

Расположение:	Переставляем элементы:
x … *y* … ***B***	***x*** и ***B***
B … ***y*** … *x*	***B*** и ***y***
y … ***B*** … ***x***	***B*** и ***x***
y … *x* … ***B***	

Таким образом, реализация алгоритма решения подзадачи 1 будет такой:

- по матрице смежности графа определяем вершину ***B***, смежную со всеми остальными вершинами;

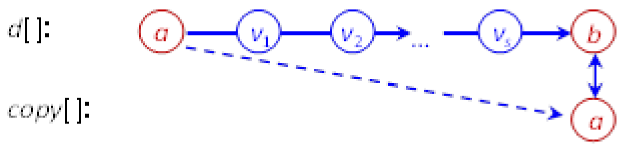
- находим вершину с наименьшим рейтингом d_1 и с помощью не более, чем *трёх* перестановок перемещаем её на первое место в массиве рейтингов d;
- затем находим вершину со следующим по величине рейтингом d_2 и ставим её на второе место указанным выше способом. И так далее...

Замечание. Можно воспользоваться алгоритмом быстрой сортировки [1, 2] исходного массива рейтингов. После этого, зная положение каждого элемента в отсортированном массиве, можно заниматься перестановкой элементов на требуемые позиции. В этом случае сложность описанного алгоритма будет иметь порядок $O(n \log n)$, где n — число вершин графа.

Подзадача 2. Создадим массив *copy*[] — копию исходного массива рейтингов и отсортируем его числа по возрастанию. Теперь мы знаем, на какое место в графе нужно переместить каждое число исходного набора.

Пусть a — произвольная вершина графа и b — вершина графа, на место которой необходимо переместить вершину a. С помощью обхода в ширину находим кратчайший путь от a до b. Если такого пути не существует, выводим –1.

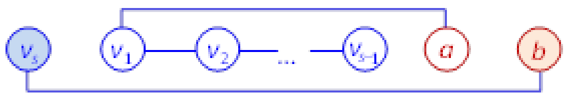
Пусть этот путь состоит из (s+1) рёбер: $a — v_1 — v_2 — ... — v_s — b$.



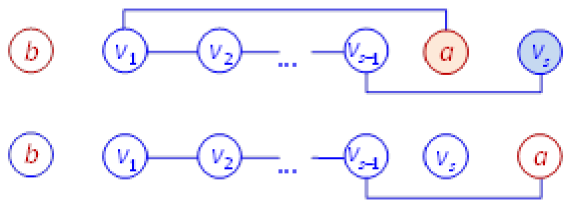
Например, если длина пути равна 1, сотрудников a и b можно пересадить за один шаг; если равна 2, — за 3 шага. С помощью индукции попытаемся оценить количество шагов. Выделим часть пути от a до v_s длиной s рёбер:

$$(a — v_1 — v_2 — ... — v_s) — b.$$

Предположим, что за k шагов можно переставить числа в вершинах a и v_s, то есть через k шагов получим следующее расположение элементов массива рейтингов:



Затем за один шаг переставим числа b и v_s (это возможно, поскольку между вершинами b и v_s есть ребро). Используя рекурсию, ещё за k шагов поменяем числа в вершинах a и v_s пути (a — v_1 — v_2 — ... — v_s) — b:



В результате вершины a и b поменялись местами, причём остальные элементы массива остались на прежних местах. В этой процедуре обмена потребовалось k+1+k=2k+1 шагов для перестановки двух чисел-рейтингов a и b. Например, для k=1 число шагов равно трём; следующее число шагов равно 2·3+1=7, для k=4 будет 2 7+1=15 шагов, и так далее.

Ситуация, связанная с подсчётом числа шагов в этой сортировке, напоминает известную задачу о «Ханойских башнях», да и окончательная формула числа шагов для пути длиной s рёбер совпадает с числом операций в этой известной головоломке: 2^s-1.

Сложность приведённого решения — $O(n^2 2^s)$, где s — максимальная длина пути между вершинами a и b ($s<n$).

Подзадача 3. Приведём теперь решение задачи, которое можно считать вполне удовлетворительным по числу операций, необходимых для сортировки массива, его алгоритмическая сложность имеет порядок $O(n^3)$.

Основная идея — циклический сдвиг чисел v_i вдоль пути между вершинами a и b. Проиллюстрируем на примере пути a — v_1 — v_2 — v_3 — b. Сначала за s=3 шага выполним циклический сдвиг вправо первых s+1=4 элементов этого пути. Это можно сделать следующим образом:

Расположение:	Переставляем элементы:
$\mathbf{a}$ — $\mathbf{v_1}$ — v_2 — v_3 — b	$\mathbf{a}$ и $\mathbf{v_1}$
$\mathbf{v_1}$ — a $\mathbf{v_2}$ — v_3 — b	$\mathbf{v_1}$ и $\mathbf{v_2}$
$\mathbf{v_2}$ a — v_1 $\mathbf{v_3}$ — b	$\mathbf{v_2}$ и $\mathbf{v_3}$
v_3 a — v_1 — v_2 b	

Теперь поменяем местами числа в вершинах v_3 и b. Это можно сделать за один шаг, поскольку между ними есть ребро.

В результате получим: $b\ a — v_1 — v_2 — v_3$. Рассмотрим путь от a до v_3 в «*обратном*» направлении, и за s=3 шага снова выполним циклический сдвиг влево s+1=4 элемента этого пути: $v_1 — v_2 — v_3\ a$. В итоге получим нужное расположение вершин a и b:

$b\ v_1 — v_2 — v_3\ a$.

В общем случае такая «циклическая» перестановка элементов реализуется за $2s–1$ шагов, где s — длина пути между вершинами a и b. Итоговая сложность алгоритма — $O(n^3)$.

ЗАКЛЮЧЕНИЕ

Как отмечалось в [3], создание качественных олимпиадных задач по математике и информатике является сложным и трудоёмким процессом. Большинство задач, которые предлагаются на таких интеллектуальных соревнованиях, представляют собой многоуровневые исследовательские проблемы. В статье представлена одна из таких оригинальных задач, постановка которой близка к классической фундаментальной задаче сортировки данных.

СПИСОК ЛИТЕРАТУРЫ

1. *Кнут Д.Э.* Искусство программирования. Том 3. Сортировка и поиск. 2-е изд. М.: Издательский дом «Вильямс», 2007, Т. 3, 832 с.
2. *Кормен Т.Х., Лейзерсон Ч.И., Ривест Р.Л., Штайн К.* Алгоритмы: построение и анализ. М.: Издательский дом «Вильямс», 2005. 1296 с.
3. *Киндер М.И.* Классические комбинаторные объекты на соревнованиях по программированию // Информационные технологии в образовании и науке. ИТОН 2016: Материалы международной научно-практической конференции. Казань: Изд-во Академии наук РТ, 2016, С. 46–52.

SORTING PROBLEM ON GRATHS IN PROGRAMMING CONTESTS

Mihail Kinder[1], Andrei Kazantsev[2]

Kazan (Volga region) Federal University

[1] *mkinder@rambler.ru*, [2] *Andrei.Kazantsev@kpfu.ru*

Abstract

The problem of sorting data is analyzed, the order relation between which is described as the adjacency relation of vertices on an arbitrary graph. Subtasks and issues related to the ‘neighborhood‘ of the problem are highlighted; their solution is the level of ‘immersion‘ in the solution of the general problem. Algorithms for solving individual subtasks for graphs of a special kind are discussed, as well as various approaches to solving the sorting problem in the general case. A sorting task of this type was proposed at the ISI-Junior School Programming Cup in July 2019 (Innopolis).

***Keywords**: mathematical olympiads, programming contests, informatics olympiads, multilevel tasks in mathematics, multilevel tasks in informatics contests, sorting problem on graphs*

REFERENCES

1. *Knut D.E.* Iskusstvo programmirovaniya. Tom 3. Sortirovka i poisk, 2-ye izd. M.: Izdatel'skiy dom «Vil'yams», 2007. T. 3. 832 s.
2. *Kormen T.X., Leyzerson Ch.I., Rivest R.L., Shtayn K.* Algoritmy: postroyeniye i analiz. M.: Izdatel'skiy dom «Vil'yams», 2005. 1296 s.
3. *Kinder M.I.* Klassicheskiye kombinatornyye ob’yekty na sorevnovaniyakh po programmirovaniyu // Informatsionnyye tekhnologii v obrazovanii i nauke. ITON 2016: Materialy mezhdunarodnoy nauchno-prakticheskoy konferentsii. Kazan': Izd-vo Akademii nauk RT, 2016, C. 46–52.

СВЕДЕНИЯ ОБ АВТОРАХ

КИНДЕР Михаил Иванович – доцент кафедры высшей математики и математического моделирования Института математики и механики им. Н.И. Лобачевского КФУ.

Mihail Ivanovich KINDER – associate professor, Department of Further Mathematics and Mathematical Modelling of N.I. Lobachevsky Institute of Mathematics and Mechanics KFU.

email: mkinder@rambler.ru

КАЗАНЦЕВ Андрей Витальевич – доцент кафедры математической статистики Института вычислительной математики и информационных технологий КФУ.

Andrei KAZANTSEV – associate professor at Institute of Computer Mathematics and Information Technologies KFU.

email: Andrei.Kazantsev@kpfu.ru

Материал поступил в редакцию 15 сентября 2019 года