Разбор задачи «Pinball»

Автор задачи: С.Астафьев, автор разбора: Е.В.Андреева

Эта задача была одной из самых простых на соревновании. Тем не менее, многие участники потеряли большое число баллов за ее решение из-за того, что упустили некоторые случаи.

При решении необходимо рассмотреть два возможных варианта. Либо шарик, пущенный из точки A, попадает в точку B и без установки дополнительной пластинки, либо это не так. Начнем рассмотрение решения со второго случая.

Построим траекторию движения шарика, выпущенного из точки A. Она либо закончится в одной из граничных клеток, либо зациклится (признаком зацикливания является повторное попадание в некоторую клетку поля, причем при движении в том же направлении, что и ранее). Для каждой клетки траектории подсчитаем расстояние от точки A (если хранить полученные значения в таблице, размер которой совпадает с размером поля, то следует учесть, что если траектория пересекает клетку как в горизонтальном, так и вертикальном направлении, то хранить следует два числа). Построим другую траекторию движения шарика, начиная с точки B, запоминая расстояния уже от этой точки. Очевидно, что отражающую пластинку надо поставить в одну из точек пересечения под прямым углом этих двух траекторий (а хотя бы одна такая точка всегда найдется, поскольку в противном случае задача не имела бы решения, что невозможно по условию). Поэтому следует перебрать все точки пересечения под прямым углом двух траекторий и выбрать ту, которая приводит к результирующему пути минимальной длины. С учетом предварительно сохраненной информации длину получающейся при установке пластины траектории можно каждый раз определять за $O(1)$. Так как длина любой траектории не может превосходить $2N^2$, общее число операций в рассмотренном решении равно $O(N^2)$.

В первом же случае переберем все клетки, через которые проходит траектория шарика, выпущенного из точки A. Если есть клетки, через которые траектория проходит два раза, то установка отражающей пластинки в такой клетки “отщепляет” от траектории цикл. Следует выбрать клетку, после установки пластинки в которой длина оставшейся траектории будет минимальна. Если такой клетки нет, можно постараться оставить длину траектории без изменения. Для этого следует поставить пластинку в свободную клетку, через которую траектория шарика не проходит. Наконец, если и такой клетки нет, то устанавливать пластинку нужно в клетку траектории так, чтобы шарик, совершив некоторый цикл, снова вернулся на траекторию. При этом следует ставить пластинку в клетку пересечения траектории с образующимся циклом наименьшей длины — это минимально увеличивает длину траектории шарика. Для эффективного решения этой задачи для каждой точки траектории нужно предварительно вычислить длину цикла, который может образоваться, если в соответствующую клетку поставить пластинку, отслеживая каждый из циклов не более одного раза. После этого длину получающейся траектории движения шарика из A в B можно опять определять за $O(1)$. Общее время решения задачи, как и ранее, имеет порядок $O(N^2)$.

Разбор задачи «Великолепный Гоша»

Автор задачи и разбора: Е.В.Андреева

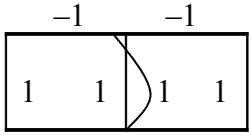
Для того чтобы понять, какая последовательность операций может привести к исправлению имеющейся сети согласно схеме, в первую очередь выясним, на сколько построенная сеть отличается от исходной схемы. Причем сделать это несложно уже в процессе считывания входных данных. Информацию о проложенных и требуемых кабелях будем сохранять в двумерном, предварительно обнуленном массиве A, состоящем из $N \times N$ элементов. При считывании описания схемы в случае наличия в схеме кабеля, соединяющего дома с номерами i и j, значения каждого из элементов массива A_{ij} и A_{ji} будем уменьшать на единицу, а если Гоша проложил кабель между некоторой парой домов, то при считывании описания сети будем увеличивать соответствующие элементы на единицу.

В результате мы получим описание так называемого “графа различий” между схемой и проложенной сетью. Вершинами этого графа будут являться N рассматриваемых в задаче домов. Если кабель между парой домов как на схеме, так и в построенной сети отсутствует, или имеется как на схеме, так и в сети, то соответствующие вершины графа ребром соединены не будут (различия в данном случае отсутствуют). Последнюю ситуацию не надо отслеживать специально. При считывании описания схемы значения соответствующих двух элементов массива A станет равным –1, а при дальнейшем считывании описания кабеля, соединившего эти же дома, те же элементы массива будет увеличены на 1 и станут равным 0 (аналогичная ситуация складывается и в случае двух, трех и более кабелей как имеющихся на схеме, так и реально проложенных Гошей между некоторой парой домов). В противном случае две вершины такого графа будут соединены одним или несколькими ребрами, помеченными либо только единицами, либо — минус единицами. Ребро окажется помеченным –1, если Гоша не проложил кабель между соответствующими домами, а на схеме он был, таких ребер будет несколько, если он проложил меньше кабелей, чем этого требовалось по схеме. Ребра, помеченные единицами означают, что кабелей в сети между данной парой домов оказалось больше, чем было необходимо. Вместо хранения каждого ребра в отдельности, мы получаем в описанном выше массиве одно положительное или отрицательное число, характеризующее как количество ребер, так и их принадлежность (схеме или сети).

Для примера, приведенного в условии задачи, массив A будет выглядеть следующим образом:

0	–1	0	1	0	0
–1	0	–1	0	2	0
0	–1	0	0	0	1
1	0	0	0	–1	0
0	2	0	–1	0	–1
0	0	1	0	–1	0

Заметим, что так как число кабелей, выходящих из любого дома, на схеме и в сети одинаково, из каждой вершины такого графа выходит обязательно четное число ребер. Более того, у любой вершины число ребер, помеченных минус единицами, совпадает с количеством “единичных” ребер. Поэтому легко показать, что полученный граф строго циклический, более того, его можно разбить на циклы вида 1 –1 1 –1 … 1 –1 (здесь приведены обозначения для ребер, из которых может быть составлен подобный цикл), возможно, несколькими способами. На рис. 3 показан граф различий для примера из условия. Положительные ребра в нем помечены сплошными линиями, а отрицательные — пунктирными.



–1 –1

Рис. 3

В нем можно выделить либо два цикла указанного вида, либо один, соединяющий вершины в следующем порядке: 1—2—5—6—3—2—5—4—1.

Для любых четырех соседних вершин цикла такого вида (ребер с метками 1 –1 1, связывающих именно четыре, а не три вершины), можно выполнять описанную в условии задачи операцию, после чего цикл либо сократится на два элемента (три ребра исчезнут и появится одно новое, помеченное единицей), но будет иметь тот же вид (так как последовательность ребер 1 –1 1 заменяется на ребро, помеченное 1), либо, если такой цикл состоял всего из 4-х ребер, он просто исчезнет). Следовательно, за одну операцию мы можем гарантированно сократить число различий на два и решить нашу задачу не более чем за M операций, вне зависимости от способа разбиения на указанные циклы. Это заведомо меньше ограничения на 48 000 операций, приведенного в условии задачи.

Обратите внимание, что для выполнения операции нам подходят не любые три смежных ребра, помеченных 1, –1 и 1 соответственно. Подобные ребра могут составлять и треугольник, а не четырехугольник, например, как это показано на рис. 4.

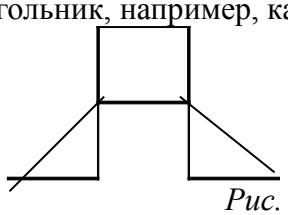


Рис. 4

В этом “графе различий”, во-первых, не все циклы имеют нужный нам вид (хотя нужный цикл тоже имеется и содержит все ребра), во-вторых, даже в нужном цикле имеются последовательности ребер, помеченных 1 –1 1, образующие треугольники, а не четырехугольники и над ними допустимую операцию выполнить невозможно. Но тогда разрешенную операцию заведомо можно выполнить, используя, например, последнее из рассмотренных ребер и некоторые два других смежных знакочередующихся ребра. Таким образом задача всегда разрешима.

В результате мы приходим к следующему простому алгоритму решения задачи.

1) Ищем в массиве A произвольный положительный элемент A_{ij}. Если таковой отсутствует, то конец алгоритма.
2) В j-й строке массива ищем отрицательный элемент A_{jk} (такой элемент обязательно найдется).
3) В k-й строке массива ищем положительный элемент A_{kl}, $l \neq i$.
4) Если такой элемент найден, то выполняем п. 5, в противном случае — п.6.
5) Уменьшаем значения A_{ij}, A_{ji}, A_{kl} и A_{lk} на единицу и, соответственно, увеличиваем на единицу значения A_{jk}, A_{kj}, A_{li} и A_{il}. Выводим соответствующую операцию ($i\ j\ k\ l\ i\ l\ k\ j$) на печать. Переходим к п. 1.
6) Переопределяем значения i и j: $j = i$; $i = k$. Переходим к п. 2.

Заметим, что во многих случаях в п. 1 алгоритма поиска элемента возможно избежать: искомым элементом может стать A_{il}, так как при выполнении предшествующей поиску операции по перестройке сети он увеличивается на 1. В любом случае пункты 2) и 3) алгоритма будут выполняться лишь для ненулевых элементов массива A, которых не более чем $2M$, а массив A для поиска ненулевых элементов можно просматривать всего один раз. Таким образом вычислительная сложность алгоритма составит $O(NM + NN)$, что укладывается в приведенные в условии задачи ограничения. Но решение можно было бы сделать еще более эффективным, в случае представления “графа различий” в виде одномерного массива списков вершин, смежных с данной.

Разбор задачи «Красная шапочка»

Автор задачи и разбора: Г.Корнеев

После формализации условия задача сводится к поиску самого быстрого пути во взвешенном графе, ребра которого могут на вермя исчезать (когда по ним пробегает Волк).
Заметим, что ребра пропадают и появляются вновь только в целые времена. Пусть существует маршрут по которому Красная Шапочка может прибежать к домику Бабушки за время $T_{кш}$. Легко показать (оставляем это вам в качестве упражнения), что тогда она сможет добежать по этому маршруту и за любое время из интервала ($[T_{кш}]$, $[T_{кш}] + 1$), где $[T_{кш}]$ — обозначает целую часть числа $T_{кш}$. Поэтому мы можем считать, что Красная Шапочка выходит из дома не в нулевой момент времени, а сначала сидит в домике время $0 < \varepsilon < 1$. После чего бежит к дому Бабушки, заходя и уходя с тропинки только в моменты времени $T_i + \varepsilon$, где T_i — целое. В этом случае она никогда не появится одновременно с Волком на полянке (когда Волк "занимает" сразу несколько тропинок). Это соображение позволяет легко формализовать процесс появления и исчезновения ребер в графе, доступных Красной Шапочке для перемещения.

"Занятость" тропинок можно отслеживать, храня для каждой тропинки промежутки времени, когда по ней не бежит Волк. При поиске кратчайшего пути в таком динамически меняющемся графе будем учитывать, что если интервал между пробеганиями Волка меньше времени, необходимого Красной Шапочке, чтобы пройти тропинку, то в этот интервал она тропинкой воспользоваться не сможет, но, возможно, имеет смысл проанализировать следующие временные интервалы, в которые данная тропинка "свободна".
Заметим, что если Красная Шапочка может оказаться на некоторой полянке в момент времени $T_1 + \varepsilon$ то она может спрятаться, а затем уйти с нее в любой момент $T_2 + \varepsilon$, при $T_1 \le T_2$. То есть, если выяснилось, что Красная Шапочка может успешно достичь i-й полянки за время T_i, то ухудшиться это время уже не может, вне зависимости от дальнейшего поведения волка. Таким образом, несмотря на то что в нашем графе ребра на некоторое время исчезают, для поиска самого быстрого пути можно воспользоваться алгоритмом Дейкстры поиска кратчайшего пути во взвешенном графе (см., например, "Информатику" № 8/2002). При этом, осуществляя релаксацию (пересчет кратчайшего пути между первой и i-й полянкой) нужно учитывать, что Красная Шапочка может пройти по тропинке, уменьшающей время достижения i-й полянки, только если она не будет "занята" Волком.

Таким образом, решение задачи сводится к реализации алгоритма Дейкстры, и определению на каждом шаге "занятости тропинки". Причем отслеживание "занятости" тропинок требуется реализовывать очень аккуратно по времени, так как путь Волка может включать до 100 000 тропинок, порождая большое число временных интервалов "занятости" каждой тропинки.

Ответ на вопрос задачи зависит от минимального времени, за которое Красная Шапочка может благополучно достигнуть домика Бабушки. Если это время окажется меньше, чем известное нам время прихода в это же место Волка, то мы получаем утвердительный ответ на вопрос задачи. Порядок прохождения тропинок Красной Шапочкой будет не сложно восстановить, если во время реализации алгоритма Дейкстры для каждой полянки запоминать тропинку, по которой Красная Шапочка непосредственно придет на данную полянку, двигаясь оптимальным способом от полянки номер один (напомним, что две полянки могут быть соединены и несколькими тропинками, а "спасительной" может оказаться лишь одна из них).