Задачи заключительного этапа Всероссийской олимпиады школьников по информатике 2010 года
(19 – 25 апреля 2010 г., Ханты-Мансийск)

Задача 1 «Работы»

Есть две работы A и B и есть n машин: M1, M2, M3, …, Mn . Для каждой работы задан порядок её обработки на машинах. Работа A имеет порядок обработки i1, i2, i3, …, ik , а работа B имеет порядок обработки j1, j2, j3, …, jl . Также для каждой операции обработки (их всего k+l) задано время обработки.

Требуется найти оптимальное расписание. В произвольный момент времени никакая машина не может обрабатывать обе работы сразу, и никакая работа не может выполняться на нескольких машинах. Каждая работа должна проходить машины в заданном порядке. Машина может разрывать обработку работы на произвольное количество кусков (делать прерывания). Целевая функция, которую требуется минимизировать, зависит от времён окончания обработки работ и может быть практически любой разумной: можно, например взять общее время обработки (max(tA + tB)), или сумму квадратов времён выполнения работ (tA2 + tB2) или что-нибудь ещё.

Рекомендации по оцениванию и решению задачи

Геометрическая интерпретация задачи: каждой точке на плоскости (x, y) сопоставляем состояние работ, при котором ровно x времени всего потрачено на обработку работы A и ровно y времени потрачено на обработку работы B. Тогда прямоугольник [0; TA] x [0; TB] – область возможных состояний (TA – общее время обработки работы A). Стартовое состояние – (0; 0), конечное – (TA; TB). Перемещаться по плоскости можно со скоростью (1; 0) – обрабатывая только первую работу, (0; 1) – обрабатывая только вторую работу и (1; 1) – обрабатывая обе работы параллельно в случае, если машины для обработки разные. Области, на которых у работ машины для обработки совпадают, являются прямоугольниками. Получается, что плоскость состояний выглядит как прямоугольное клеточное k x l с клетками произвольного размера, причём некоторые клетки запрещены для одновременной обработки.

Чтобы найти оптимальное решение, будем искать минимальное время, необходимое для того, чтобы добраться до каждого из (k+1) x (l+1) узлов поля. Для нахождения расстояний обходим все узлы в лексикографическом порядке, и из каждого пробуем построить некоторые пути: мы идём со скоростью (1; 1) по диагонали, пока не случится одно из:

1. Попадание в узел сетки: обновляем мин. время до этого узла и прекращаем построение пути.
2. Выход на верхнюю или правую границу: в этом состоянии одна из работ выполнена. Считаем времена окончания обработки работ tA и tB и сравниваем целевую функцию для них с текущим лучшим ответом задачи. Построение пути прекращаем.
3. Пересечение границы запрещённого прямоугольника: в этом случае дальше обрабатывать обе работы нельзя. Есть два варианта – идти вправо или вверх (обрабатывать только работу A или только работу B). В одном из вариантов мы будем двигаться вдоль границы прямоугольника и придем в узел. Обновляем мин. время для этого узла. Во втором случае мы проходим сквозь запрещённую зону. После выхода из неё мы продолжаем строить путь по диагонали.

Построение одного пути занимает не более k+l времени, поэтому общее время работы – O(kl(k+l)).

Задача 2 «Вершинное покрытие»

Дан неориентированный граф, содержащий n вершин. Требуется найти его вершинное покрытие, содержащее не более k вершин, либо выяснить, что такого не существует. $n \leq 1000$, $k \leq 10$.

Вершинное покрытие — это множество вершин, такое что каждое ребро графа имеет хотя бы один конец в этом множестве.

Рекомендации по оцениванию и решению задачи

Очевидно переборное решение за n^k. Однако забавно, что у этой задачи есть решение за полином от n при любом фиксированном k. Коэффициент, разумеется, растет экспоненциально от k.

Например, решение за n * k * 2^k. Возьмем ребро. Один из его концов лежит в В.П., переберем, какой. Удалим эту вершину и инцидентные ей ребра из графа и повторим. Если за k не удастся уничтожить весь граф, то ответ «нельзя». Иначе нашли В.П. Сложность каждого шага порядка n, а глубина перебора k, поэтому время работы n * k * 2^k. Техническая трудность (чтобы не получилось n^2 * k * 2^k) в выборе произвольного ребра.

Кажется, простая задача на перебор, но к перебору надо подойти с неочевидной стороны.

Задача 3 «Серверы 1D»

Компьютерная сеть в некотором доме строилась по принципу присоединения нового компьютера к последнему из уже подключенных. Никакие два компьютера, будучи подключенными в сеть, между собой проводами дополнительно не связывались. Таким образом в сеть были объединены N компьютеров. Соседи обменивались информацией между собой, но в какой-то момент поняли, что им не хватает серверов. Они решили некоторые компьютеры сделать прокси-серверами. Компьютерное сообщество дома имеет возможность установить ровно K серверов. Осталось только решить, какие именно компьютеры будут прокси-серверами. Главным критерием является ежемесячная стоимость обслуживания серверами всех компьютеров.

Для каждого компьютера установлен тариф его обслуживания, выраженный в рублях за метр. Стоимость обслуживания одного компьютера каким-то сервером равна тарифу компьютера, умноженному на длину провода от этого компьютера до сервера, которым он обслуживается.

Ваша задача написать программу, которая так расставит K серверов, чтобы общие затраты на обслуживание всех компьютеров были минимальными.

Входные данные

В первой строке входного файла записано два целых числа N и K ($K \leq N$)— количество компьютеров в сети и количество серверов, которые нужно установить.

Все компьютеры в сети пронумерованы числами от 1 до N по порядку подключения.

Во второй строке записано одно целое число — тариф обслуживания первого компьютера.

В следующих N – 1 строках записано через пробел по 2 целых числа Li, Ti — информация об остальных компьютерах в сети по порядку номеров. Li – длина провода, соединяющего i-компьютер с соседним с меньшим номером, Ti — тариф обслуживания данного компьютера ($2 \leq i \leq N$).

Выходные данные

В первую строку выходного файла необходимо вывести одно целое число — минимальную стоимость обслуживания всех компьютеров всеми серверами. Во второй строке должны быть записаны через пробел номера компьютеров, в которых надо разместить серверы.

Рекомендации по оцениванию и решению задачи

Динамика за O(NK), с предподсчетом частичных сумм взвешенных расстояний за O(N).

Частичные решения:

1) Перебор
2) K=1 (мб и для K=2)
3) Динамика за O(N2K) — используется принцип решения как для дерева, с использованем 2-х рекурентных формул, и Min в каждой итерации для каждой вершины считается заново с перебором всех разбиений отрезка на две части, без учета результатов для предыдущей вершины, т.е. без использования свойства линейности сети.
4) Сама динамика за O(NK), но предподсчет частичных сумм взвешенных расстояний в лоб за O(N2).

Задача 4 «Сериал»

Антон смотрел сериал Lost. В некоторый момент у него возникли неотложные дела, и через месяц, когда он решил досмотреть его, он уже не помнил, на какой серии он остановился (тому виной не самый тривиальный сюжет). Скачанные файлы у него не сохранились. Он делает следующим образом: скачивает серию за *A* минут, после чего мгновенно определяет, смотрел ли он ее. Если да, то он оставляет ее на диске. Цель: найти место, с которого надо продолжить просмотр и досмотреть сериал за минимальное время. На просмотр одной серии уходит *B* минут.

Если действовать по оптимальной стратегии, то:

- За какое время Антон управится, если на самом деле он просмотрел первые *K* серий?
- Какое время у него уйдет в среднем?

Рекомендации по оцениванию и решению задачи

Задача решается методом динамического программирования:

1. Решение за $O(N^3)$. Состояние --- отрезок неизвестных серий. То есть f[l, r] – наименьшее время, которое необходимо потратить, чтобы Антон смог найти неоходиму серию, если ему достоверно известно, что первая непросмотренная серия находится в отрезке [l, r]. При этом известно, что серия r – заведомо непросмотрена. Ответ f[1, n + 1]. Начальное состояние f[i,i] = 0. Переход:
   $$f[L, R] = \min_{i=\{L, R-1\}}(\max(f[i+1, R], f[L, i] + B))$$
2. Решение за $O(N^2)$. Можно заметить, что конкретные L и R не так важны. Важна лишь длина отрезка. g[i] = f[L, L + i]. Тогда начальное состояние g[0] = 0. Ответ g[n + 1]. Переходы: $g[R] = \min_{i=\{1,R-1\}}(\max(g[R-i], g[i] + B))$
3. Решение за O(NlogN). Формулы такие же как во втором решение. Отличие лишь в быстром пересчете переходов. Рассмотрим внимательно внутренний максимум max(g[R-i], g[i] + B). Разложим формулу на два случая, а именно

$g[R] = MIN(\min_{g[R-i]>=g[i]+B} (g[R-i]), \min_{g[R-i]<g[i]+B} (g[i]+B))$

Таким образом, мы избавились от внутреннего максимума и свели к рассмотрению двух случаев. Нет сомнений, что функция g[i] возрастающая, а g[R-i] соответственно убывающая. Тогда найдем все i, при которых g[R-i] >= g[i] + B. g[i] – g[R-i] <= -B. Функция g[i] – g[R-i] возрастающая, поэтому неравенство выполнено для 1 <= i < i'. i' можно найти бинарным поиском.

Итого получаем, что $g[R] = MIN(\min_{1 <= i < i'} g[R-i], \min_{i' <= i < R} g[i] + B)$

g[R-i] --- убывает, поэтому первый минимум равен значению при наибольшем индексе i. Второй минимум аналогично равен значению при наименьшем индексе.

Задача имеет довольно много частичных решений, чем довольно интересна. Так же вторая часть задачи по нахождению количества необходимых ходов в случае, если известно, что он просмотрел ровно K серий --- придает изюминку задаче. На мой взгляд, не стоит просить находить среднее значение, а лучше попросить вывести ответ для всех K.

Частичные решения сильно отличаются асимптотически и легко разделяются тестами. Тест состоит почти из одного числа, поэтому подготовка, по сути, состоит только в написании решений.

Задача 5 «Прямоугольник»

Задана прямоугольная 0-1 матрица $n \times m$. Разрешено сделать один ход, выбрав некоторый подпрямоугольник (возможно и весь данный) и реверснуть на нем все значения.

После того, как ход сделан, в новой матрице ищется наибольший по площади прямоугольник из единиц. Какой максимальный прямоугольник из единиц может быть получен? Какой ход для этого надо сделать?

Рекомендации по оцениванию и решению задачи

Конечно, для решения этой задачи надо знать как решать классический вариант без хода. Плюсом задачи является то, что дети могут придумывать алгоритмы совсем разной эффективности от O(n^много * m^много) до O(n*m).

Возможно есть решение за O(n*m).

Задача 6 «Остров Меркет»

Остров Меркет представляет собой прямоугольник и задан координатами противоположных вершин – (0,0) – (W,H). Маяк – прямоугольник строго внутри острова, задан координатами противоположных вершин (xl,yl) – (xr,yr). Маяк и остров параллельны осям координат. Построить границу между Финляндией и Швецией – простую ломаную, чтобы:

1) Граница начиналась на южном побережье, а заканчивалась – на северном.
2) Западное побережье было целиком шведское, восточное – финское. Граница не может иметь с боковыми сторонами общих точек, кроме вершин прямоугольника.
3) Маяк целиком лежал на финской половине (маяк может иметь общие точки с границей).
4) Площади обеих частей совпадали.
5) Координаты вершин границы должны быть целыми.

Рекомендации по оцениванию и решению задачи

Задача основана на использовании идей конструктивной геометрии. Уровень сложности этой задачи – выше среднего. В процессе ее решения необходимо аккуратно разобрать все случаи, включая случай, когда ответа нет, что вполне не очевидно, и построить для них красивые картинки. При этом следует постараться сделать так, чтобы число этих случаев было как можноменьше.