Разбор задачи Конвейер

Так как количество микрокоманд в команде не больше количества конвейеров, и каждая следующая команда выполняется позже предыдущей (т.к. она должна выполняться не раньше предыдущей, а одновременно с ней она начаться не может из-за наличия хотя бы одной исключительной микрокоманды), то очевидно, что на время начала выполнения следующей команды не влияет время окончания выполнения последней команды перед текущей на этом конвейере. Оно зависит только от типа микрокоманд, выполняющихся на последних K – 1 тактах.

Допустим мы хотим начать выполнение новой команды с i-го такта. Тогда нужно убедиться, что на K тактах, начиная с i-го не будет выполняться две исключительные микрокоманды одновременно. Можно хранить конфигурацию этих K тактов в виде двоичного числа, j-й бит которого равен 1, если на j-м такте, начиная с i-го, выполняется исключительная микрокоманда. Тогда, если аналогичным образом хранить конфигурацию одной команды, проверка на возможность выполнения текущей команды с i-го такта сводится к проверке на 0 результата операции двоичного И конфигураций тактов и команды.

Таким образом, если просто выполнять то, о чем говорится в условии задачи получим решение с оценкой O(NK), которое получит примерно 50 баллов.

Улучшить это решение можно следующим образом. Так как конфигурация тактов – двоичное число из K битов, существует всего 2^K различных конфигураций. А так как количество команд может быть значительно больше 2^K, то очевидно, что начиная с какой-то команды эти конфигурации начнут повторяться, причем начнется это не позже, чем на 2^K - й команде. Таким образом можно проводить моделирование до этого момента, запоминая время окончания выполнения каждой команды, а затем, используя цикличность получить результат. Это решение и подразумевалось в качестве полного.

Разбор задачи Перестановки

Определение. Циклом перестановки A называется такая последовательность чисел x, A(x), A(A(x)) ... x.

Любую перестановку можно однозначно разложить на непересекающиеся циклы.

Выпишем теперь данную по условию задачи перестанвку A, а под ней будем выписывать перестановки B и C:

Позиция	1	...	A(1)	...	A(x)	...	N
A	A(1)	...	A(A(1))	...	A(A(A(1)))	...	A(N)
B	x	...	A(x)	...	A(A(x))	...	...
C	A(x)	...	A(A(x))	...	A(A(A(x)))	...	...

Понятно, что циклы A(...A(1)) и A(...A(x)) должны иметь одинаковую длину. Отсюда следует, что перестановки A и B должны состоять из одинаковых циклов.

Посчитаем теперь количество перестановок B с набором циклов, совпадающим с A. Каждому элементу в A, задающему цикл можно поставить в соответствие любой из элементов любого цикла той же длины. Отсюда следует следующая формула для количества циклов:

$$R=\sum_{i=1}^{N} i^{c_i}\cdot c_i!$$, где c_i – количество циклов длины i в перестановке A.

Разбор задачи Дороги

Определение. Мостом в неориентированном графе называется ребро, после удаления которого увеличивается количество компонент связности.
Очевидно, что если в неориентированном графе ориентировать мост, то из некоторые вершины будут недоступны из других. Следовательно ориентировать мост нельзя.
Таким образом первым шагом будет определение мостов, для чего можно использовать известный алгоритм, основанный на поиске в глубин.
Все остальные ребра можно ориентировать так, чтобы доступность вершин не изменилась. Это можно сделать, например, так: запустим поиск в глубину и будем ориентировать ребра в том направлении, в котором он по ним прошел. Обратные ребра ориентируются также в направлении от текущей вершины.

Разбор задачи Шары

Решение основано на динамическом программировании.
Посчитаем количество последовательностей, которые можно получить, выбрав j шаров из шаров первых i цветов:

$$d_{i,j} = \sum_{t=1}^{min(a_i, j)} d_{i-1, j-t} C_j^t \ ,$$

где a_i – количество шаров i-го цвета,
C_j^t – количество способов расставить t предметов на j местах.
Посчитаем данную величину для всех i = 1..M и j = 1..K. Тогда ответом к задаче будет $d_{M,K}$.

Разбор задачи Прямые

Прямое решение задачи за O(N^3) следует из условия: переберем все пары точек, проведем через них прямую и посчитаем, сколько точек на ней лежит. Такое решение получило бы около 50 баллов.
Однако можно в 6 раз ускорить данное решение и оно получит 100 баллов. Будем перебирать все тройки точек $1 \le i < j < k \le N$. Тогда если все они лежат на одной прямой, каждая такая тройка добавляет к ответу 3 (так как фактически мы получаем по одной точке для трех прямых ij, jk, ik). Асимптотически более правильным следует считать решение за $O(N^2 \log N)$ или за $O(N^2)$.
Переберем все точки. Для каждой из них посчитаем количество точек, лежащих на прямых проходящих через нее. Это можно сделать, например, так. Переместим данную точку в начало координат, соответственно, отнимем ее координаты от координат остальных точек. Тогда любая прямая, проходящая через эту точку однозначно определяется угловым коэффициентом. Для каждого углового коэффициента посчитаем количество прямых имеющих данный угловой коэффициент. Допустим это количество равно M_i для i-го коэффициента. Тогда количество точек, лежащих на прямых с заданным коэффициентом равно $M_i(M_i - 1)$. Сумма по всем i даст ответ для заданной точки. А сумма ответов для всех точек и есть ответ к задаче.
Посчитать M_i можно за O(NlogN), за один проход по всем угловым коэффициентам, отсортировав их. Однако, это можно сделать и за линейное время, используя тот факт, что координаты точек по модуль не превышают 10^6, откуда следует, что угловые коэффициенты прямых будут иметь вид A/B, где A и B – целые числа, по модулю не большие 10^6.

Разбор задачи Полимино

Очевидным методом решения является перебор с отсечениями. Однако простой его вариант получит около 80 баллов, так как, например в случае с доской 5x5 и 11 полимино 1x2 и одним “уголком” количество вариантов – 61950873600. Для полного решения нужно учитывать факт наличия одинаковых по форме полимино. То есть решение сводится к следующему: посчитаем количество укладок, не учитывающих цвет полимино и затем умножим результат на количество способов раскраски.