Студент: Нуриев Наиль
Группа: 163-1
Дата: 30 марта 2017 г.

Билет 48. Динамическое программирование на подотрезках, поддеревьях и подмножествах (с примерами задач).

Динамическое программирование на подотрезках.
Это класс динамики, в котором состояние – это границы подотрезка какого-нибудь массива. Суть в том, чтобы подсчитать ответы для подзадач, основывающихся на всех возможных подотрезках нашего массива. Обычно перебираются они в порядке увеличения длины, и пересчёт основывается, соответственно на более коротких отрезках.
Пример: Нахождение длины наидлиннейшего подпалиндрома строки, т.е длина максимальной по длине строки-палиндрома, получаемой удалением некоторого количества символов исходной строки, возможно нулевого.
Решение:
Воспользуемся динамикой по подотрезкам, $f(i,j)$ – длина максимального подпалиндрома на отрезке $[i;j]$ исходной строки.
База: $f(i,j) = 1, \forall i = j,\ f(i,j) = 0, \forall j < i$ Переход: Будем увеличивать размер рассматриваемого отрезка. Теперь хотим посчитать ответ для $f(i,j)$, может быть два случая: когда $s[i] \neq s[j]$, тогда просто возьмем максимум из двух вложенных отрезков, меньших размером на 1, и когда $s[i] = s[j]$, тогда помимо того максимума нужно еще взять максимум учитывая вложенный отрезок $[i+1; j-1]$ с двумя буквами по краям($s[i], s[j]$), длина такой строки, очевидно, равна $f(i+1, j-1)+2$. Итоговая формула выглядит так:

$$f(i,j) = \begin{cases} max(f(i+1,j), f(i,j-1)), s[i] \neq s[j] \\ max(max(f(i+1,j), 2 + f(i+1,j-1)), f(i,j-1)), s[i] = s[j] \end{cases}$$

Понятно, что ответом будет $f(0, n-1)$, где n – длина строки(нумерация с нуля).

Динамическое программирование на поддеревьях.
Параметром состояния динамики по поддеревьям обычно бывает вершина, обозначающая поддерево, в котором эта вершина – корень. Для получения значения текущего состояния обычно нужно знать результаты всех своих детей. Чаще всего реализуют лениво – просто пишут поиск в глубину из корня дерева.
Пример: Дано подвешенное дерево, в листьях которого записаны однобитовые числа – 0 или 1. Во всех внутренних вершинах так же записаны числа, но по следующему правилу: для каждой вершины выбрана одна из логических операций: «И» или «ИЛИ». Если это «И», то значение вершины – это логическое «И» от значений всех её детей. Если же «ИЛИ», то значение вершины – это логическое «ИЛИ» от значений всех её детей.
Требуется найти минимальное количество изменений логических операций во внутренних вершинах, такое, чтобы изменилось значение в корне или сообщить, что это невозможно. Решение:
1) Состояние динамики: $d[v][x]$ — количество операций, требуемых для получения значения x в вершине v. Если это невозможно, то значение состояния – $+inf$.
2) Начальные значения: для листьев, очевидно, что своё значение можно получить за ноль изменений, изменить же значение невозможно, то есть возможно, но только за $+inf$ операций.
3) Формула пересчёта: Если в этой вершине уже значение x, то ноль. Если нет, то есть два варианта: изменить в текущей вершине операцию или нет. Для обоих нужно найти оптимальный вариант и выбрать наилучший.

Если операция «И» и нужно получить «0», то ответ это минимум из значений $d[i][0]$, где i – сын v.
Если операция «И» и нужно получить «1», то ответ это сумма всех значений $d[i][1]$, где i – сын v.

Если операция «ИЛИ» и нужно получить «0», то ответ это сумма всех значений $d[i][0]$, где i – сын v.
Если операция «ИЛИ» и нужно получить «1», то ответ это минимум из значений $d[i][1]$, где i – сын v.

4) Порядок пересчёта: легче всего реализуется лениво – в виде поиска в глубину из корня.
5) Ответ — $d[root][value[root] \oplus 1]$.
$\oplus$ – побитовый xor.

Динамическое программирование на подмножествах.
Пример: Задан взвешенный (веса рёбер неотрицательные) граф G размера N. Найти гамильтонов цикл (цикл, проходящий по всем вершинам без самопересечений) минимального веса.
Решение:
Так как мы ищем цикл, проходящий через все вершины, то можно выбрать за «начальную» вершину любую. Пусть это будет вершина с номером 0.
1) Состояние динамики: $dp[mask][v]$ — путь минимального веса из вершины 0 в вершину v, проходящий по всем вершинам, лежащим в $mask$ и только по ним.
2) Начальные значения: $dp[1][0] = 0$, все остальные состояния изначально – $+inf$.
3) Формула пересчёта: Если i-й бит в $mask$ равен 1 и есть ребро из i в v, то:
$dp[mask][v] = min(dp[mask][v], dp[mask - (1 << i)][i] + w[i][v])$
Где $w[i][v]$ – вес ребра из i в v.
4) Порядок пересчёта: можно написать ленивую динамику, а также можно перебирать маску в порядке увеличения.
5) Ответ лежит в $d[(1 << N) - 1][0]$.