Задачи оптимизации. Во всех предыдущих задачах на структуры данных мы работали, оптимизируя какую-либо очевидную задачу (например, сумму на отрезке). Теперь же мы будем решать задачи, которые непонятно как решать, кроме как полным перебором всех вариантов ответа.

В качестве примера возьмем задачу *subset sum*. В ней нам нужно найти подмножество заданного множества с фиксированной суммой S. Если перебрать все подмножества, и посчитать сумму по каждому подмножеству.

Перебором можно решить любую задачу, если перебрать все возможные ответы (множество вещественных чисел для нас тоже конечно!).

Подмножества хочется как-то пронумеровать. К сожалению, непонятно как закодировать 2^n чисел, если раньше мы разрешали в RAM-модели числа до $C^k \cdot n^k \cdot A$. Поэтому мы просто уточним RAM-модель, и разрешим $C^k \cdot t(n)^k \cdot A$ (Считая $t(n)$ таким временем работы, что внутри нет длинной арифметики).

Динамическое программирование. Иногда бывает полезно запоминать промежуточные величины перебора. Более того, часто перебор можно «ужать», если нам в переборе нужны не все величины (например, в задаче «subset sum» достаточно помнить только общую сумму, если перебирать элементы по очереди).

Сделаем $dp(i, x) \in \{0, 1\}$, которая будет говорить, можно ли набрать сумму x с помощью первых i элементов. Тогда $dp(i, x)$ можно пересчитать через $dp(i-1, x)$ и $dp(i-1, x-w_i)$.

Требования к нашей динамике:

- Граф вычислений ацикличен.
- «Состояния» динамики явно задают нам всю необходимую информацию.

Задача о рюкзаке. Пусть нам заданы n, S, w_i, c_i (то есть элементы с весами и стоимостями). Мы хотим выбрать некоторое подмножество с суммарным весом не более S и максимальной суммой стоимостей.

Мы можем сделать динамику $dp(i, w, c) \in \{0, 1\}$, которая решит нашу задачу. Но заметим, что наше решение монотонно по параметру c (то есть, для равных i и w стоит отдавать предпочтение ответу с максимальным c). Тогда c можно сделать *значением* динамики. То есть, пересчитывать динамику $dp(i, w) \in C$ как максимум из $dp(i-1, w)$ и $dp(i-1, w-w_i) + c_i$. Кстати, заметим, что тут задача монотонна по всем параметрам сразу.

Задача коммивояжера (TSP). Заданы точки на плоскости. Надо найти кратчайший кольцевой маршрут, проходящий по всем точкам хотя бы единожды.

Есть очевидное решение за $O((n-1)!)$. Воспользуемся ДП по подмножествам, основная идея которого — понять, что нам в состоянии важнее всего только то, в каком *множестве* вершин мы уже были, и в каких вершинах мы уже оказались. Закодировать множество мы можем с помощью двоичной маски. Решение с такой идее отработает уже за $O(2^n n^2)$.

ДП по подстрокам. Отдельный трюк, когда подстроки пересчитываются через свои подотрезки. Важное отличие в том, что мы можем пересчитываться через несколько задач сразу ($dp(l, r) = dp(l, k) + dp(k, r)$), а за счет этого порядок пересчета на графе может быть неочевидным.