Динамическое программирование: обзор

Александр Куликов

Онлайн-курс «Алгоритмы: теория и практика. Методы»
http://stepic.org/217

Общие принципы

- Проанализировать структуру оптимального решения, чтобы определить подзадачи и рекуррентное соотношение на них.
- Преобразовать рекуррентное соотношение в рекурсивный алгоритм, обернув его мемоизацией и решая подзадачи сверху вниз, от бóльших к меньшим.
- Преобразовать рекурсивный алгоритм в итеративный, решая подзадачи снизу вверх, от меньших к бóльшим.
- Доказать оценку на время работы и используемую память.
- Восстановить оптимальное решение.
- Проанализировать структуры таблицы и рекурсивных вызовов, чтобы понять, нельзя ли сэкономить память.

Подзадачи: обзор

1. Числа Фибоначчи: $D[i] = F_i$.
2. Наибольшая возрастающая подпоследовательность: $D[i]$ — длина НВП, заканчивающейся в $A[i]$.
3. Расстояние редактирование: $D[i, j]$ —расстояние между $A[1 \dots i]$ и $B[1 \dots j]$.
4. Рюкзак: $D[w]$ — оптимальная стоимость заполнения рюкзака вместимости w.
5. Перемножение последовательности матриц: $D[i, j]$ — стоимость вычисления $A_i \times \cdots \times A_j$.
6. Независимые множества в деревьях: $D[v]$ — максимальный вес независимого множества в поддереве с корнем в v.

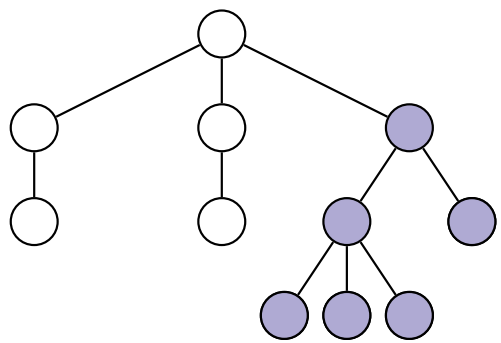
Часто используемые подзадачи

- Вход — $A[1 \ldots n]$, подзадача — $A[1 \ldots i]$.

- Вход — $A[1 \ldots n]$, подзадача — $A[i \ldots j]$.

- Вход — $A[1 \ldots n]$ и $B[1 \ldots m]$, подзадача — $A[1 \ldots i]$ и $B[1 \ldots j]$.

- Вход — дерево, подзадача — поддерево.



Снизу вверх или сверху вниз

- Преимущества решения подзадач снизу вверх:
  - нет накладных расходов на рекурсию
  - потенциальная возможность сэкономить память
- Преимущества решения подзадач сверху вниз:
  - может оказаться быстрее, если не нужно решать все подзадачи
  - неявный порядок на подзадачах