

Algorithms and data structures

lecture #7. Dynamic programming

Mentor: <....>

lecture #7. Dynamic programming

- Dynamic programming
 - Что это?
 - Для чего?
 - Мемоизация
 - Табуляция
 - Примеры
 - Выводы

Что такое Dynamic programming

- Динамическое программирование в основном представляет собой оптимизацию простой рекурсии.
- Везде, где мы видим рекурсивное решение с повторными вызовами одних и тех же входных данных, мы можем оптимизировать его с помощью динамического программирования.
- Идея состоит в том, чтобы просто хранить результаты подзадач, чтобы нам не приходилось пересчитывать их позже, когда это потребуется.
- Результат = снижает сложность времени с экспоненциальной до полиномиальной.



Начнем и закончим с Фибоначчи

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, ...


Классическая рекурсия

```
int fibonacci(int n) {  
    if (n < 2) return 1;  
    return fibonacci(n - 1) + fibonacci(n - 2);  
}
```

Исходная задача разбивается на более мелкие подзадачи, которые проще решить

То есть `fibonacci(5)` — это пятое число Фибоначчи. Чтобы посчитать пятое число Фибоначчи, нужны третье `fibonacci(3)` и четвертое `fibonacci(4)` числа Фибоначчи.

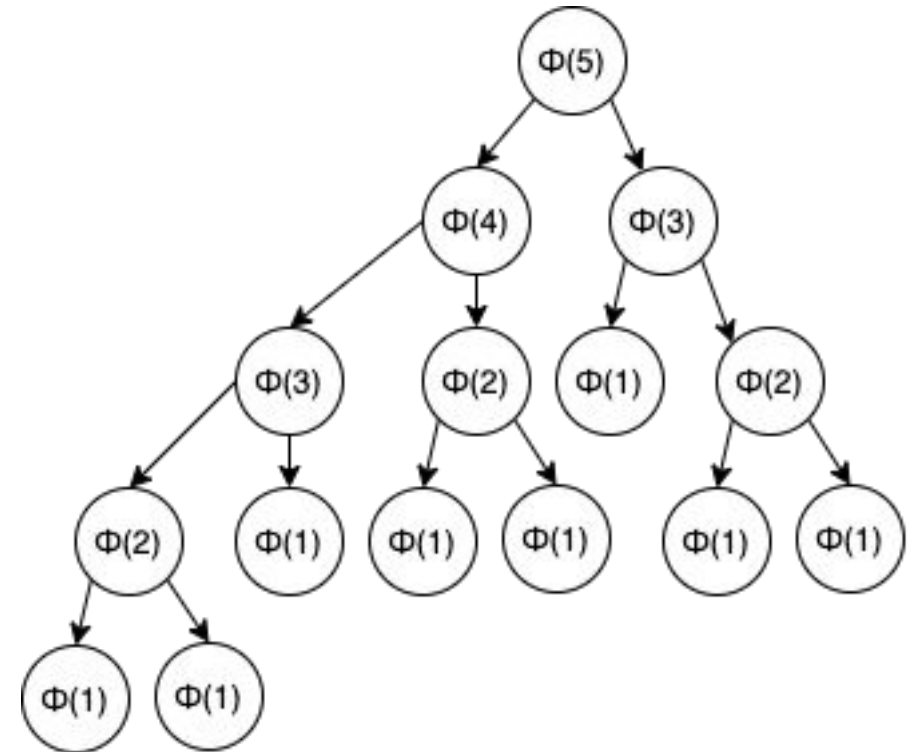
Чтобы сложить два числа ($2 + 3 = 5$), нужно меньше операций, чем, чтобы сложить 3 числа ($2 + 3 + 4 = 9$).



(Memoization) Метод мемоизации — динамическое программирование сверху вниз

- Позволяет запоминать результаты вычислений и потом переиспользовать их тогда, когда нужно сделать такие же вычисления.
- Сверху-вниз
- 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, ...

```
int fibonacci (int n) {  
    if (arr[n] != -1) return arr[n];  
    if (n < 2) return 1;  
    arr[n] = fibonacci(n - 1) + fibonacci(n - 2);  
    return arr[n];  
}
```



(Tabulation) Метод табуляции – динамическое программирование снизу вверх

- Позволяет запоминать результаты вычислений и потом переиспользовать их тогда, когда нужно сделать такие же вычисления.
- Снизу-вверх

```
arr[0] = 0;  
arr[1] = 1;  
for (i = 2; i < n; i++) arr[i] = arr[i - 1] + arr[i - 2];
```