会津大学競技プログラミング合宿2015 3日目 D問題

みゃんぱすーれつ

Myampus Sequence

- 原案: 田中
- 問題文: 田中
- 解答: 青木, 井上, 鈴木, 田中
- 解説スライド: 田中

問題

問題概要

- プログラムと数列 $x_1, \ldots, x_N$ が与えられる
- プログラムは 1 番目の関数を呼ぶ
- i 番目の関数は次のどちらかを行う
  - a_i を出力する
  - b_i 番目と c_i 番目の関数を呼ぶ
- どちらを行うかは毎回選べる
- プログラムはその数列を出力することができるか?

サンプル1

- 数列: $3, 5, 3$
- $f_1() \to 5$ を出力か $f_2(), f_3()$
- $f_2() \to 3$ を出力か $f_3(), f_1()$
- $f_3() \to 7$ を出力か $f_1(), f_2()$

$$
\begin{aligned}
f_1 &\to f_2, f_3 \\
&\to 3, f_3 \\
&\to 3, f_1, f_2 \\
&\to 3, 5, f_2 \\
&\to 3, 5, 3
\end{aligned}
$$

制約

- 数列の長さ: $1 \leq N \leq 100$
- 数列の要素: $0 \leq x_i \leq 9$
- 関数の個数: $1 \leq M \leq 10$

簡単な解説

1. 文脈自由文法の構文解析問題
2. 文法はチョムスキー標準形で与えられる
3. CYK アルゴリズム
4. $\mathcal{O}(N^3 M)$

詳しい解説

方針

1. 全探索すれば解けそう
   - 解けるとは言っていない (TLE)
   - 探索途中で同じ計算をしている気がする
2. 結局やるのは DP なんな〜

解説で使う表記

i 番目の関数が整数 x を出力することを

$$f_i \to x$$

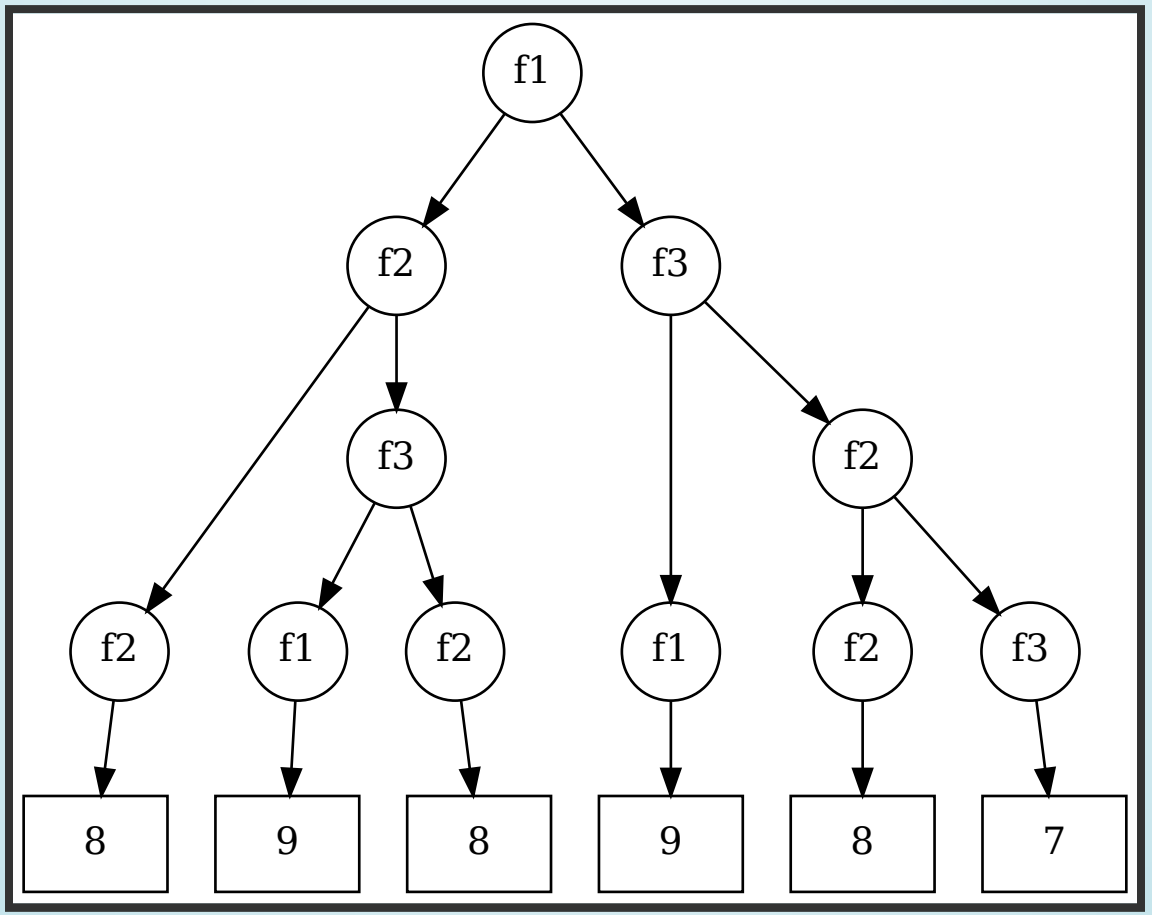
i 番目の関数が j 番目と k 番目の関数を呼ぶことを

$$f_i \to f_j\ f_k$$

と表記する

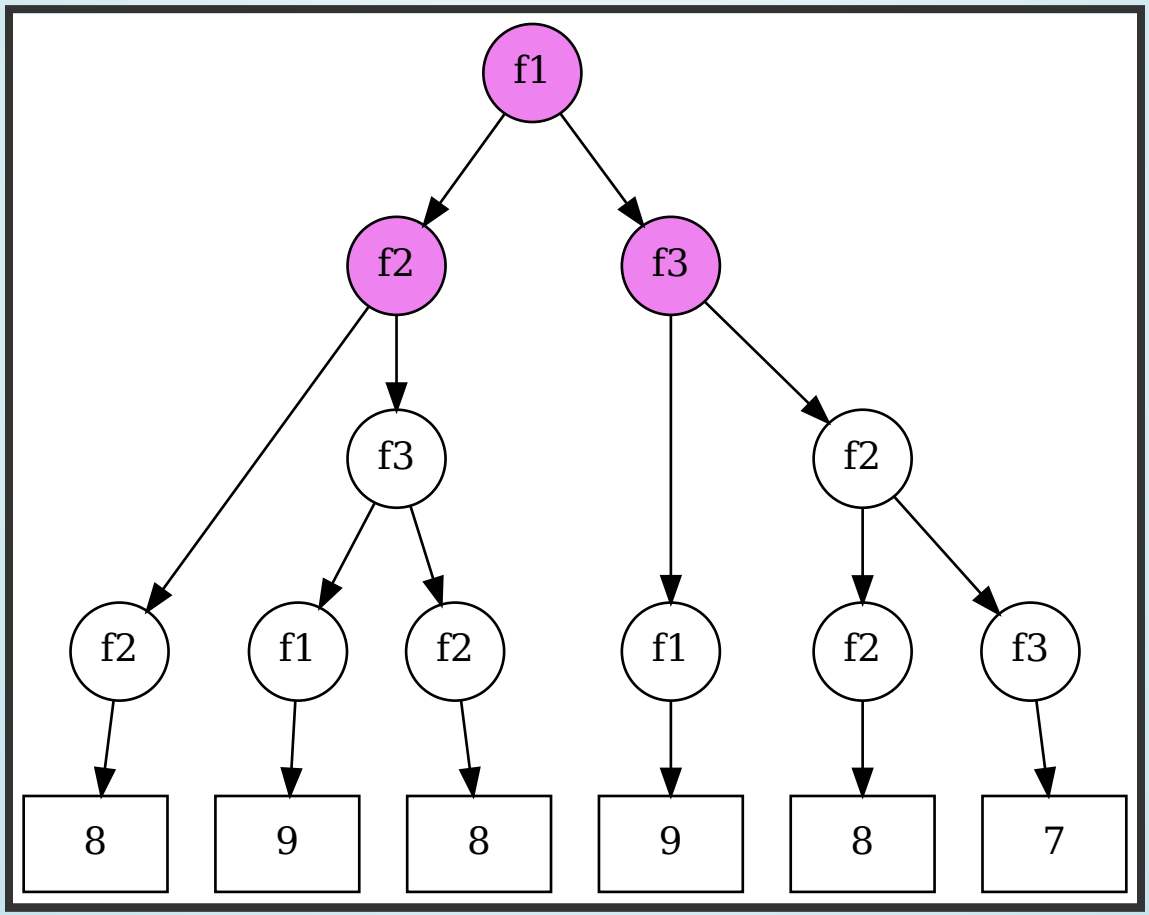
可視化

プログラムが数列を生成する様子をグラフで表す



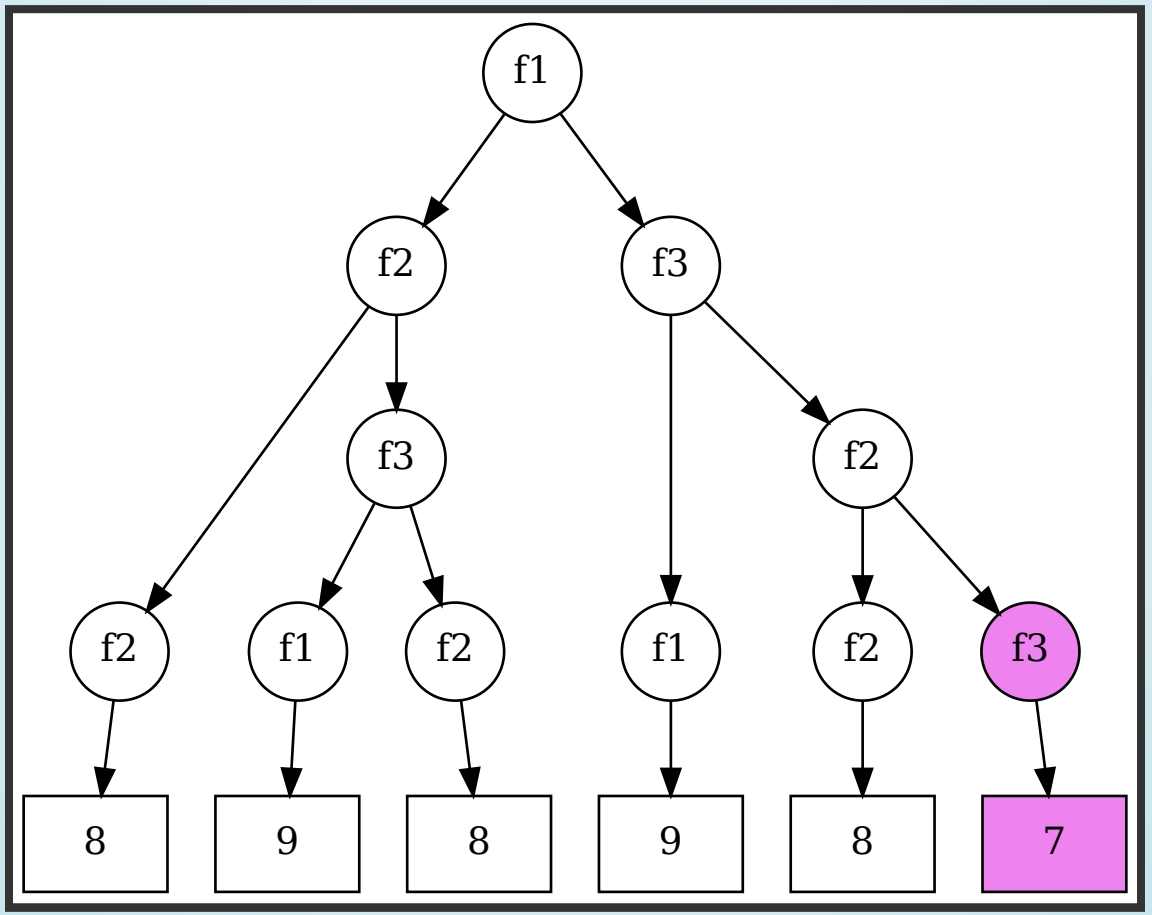
可視化

$f_1 \to f_2\ f_3$ を表す



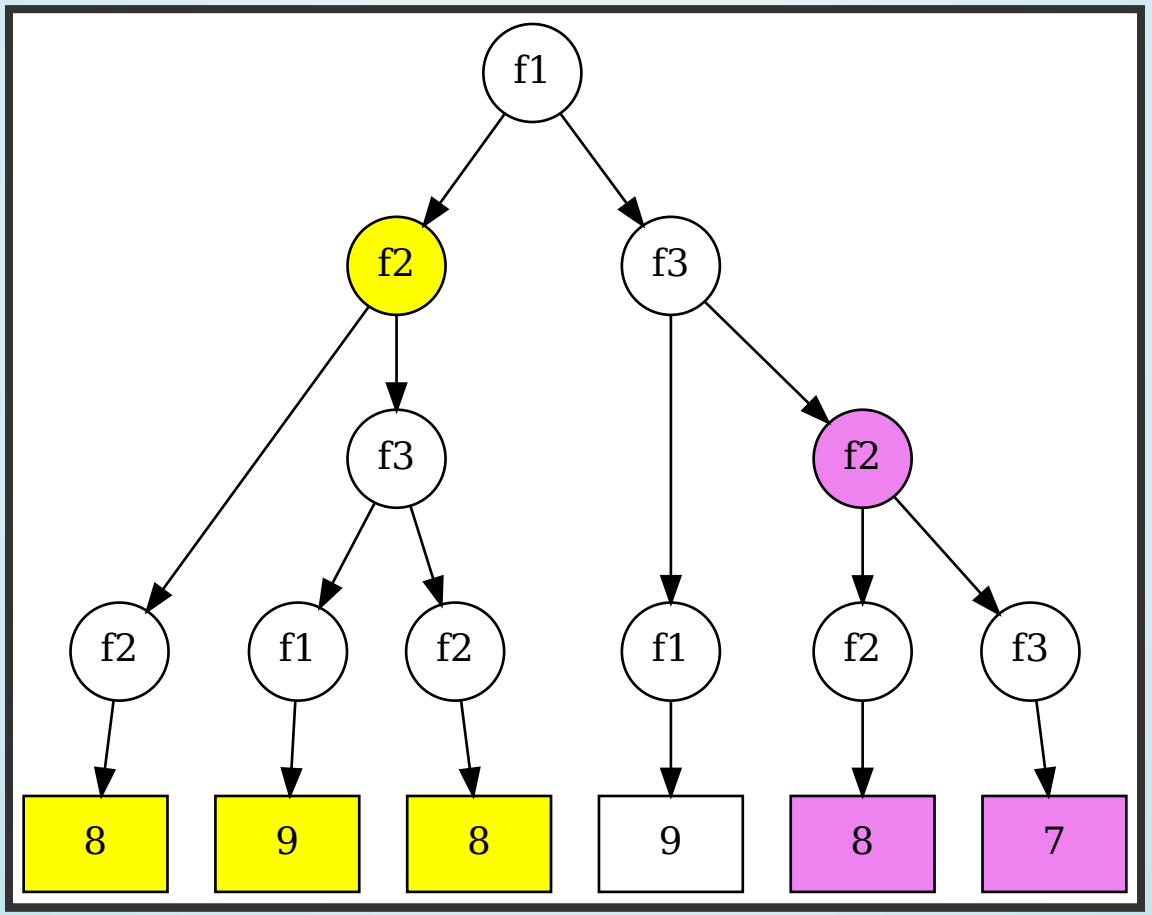
可視化

$f_3 \to 7$ を表す



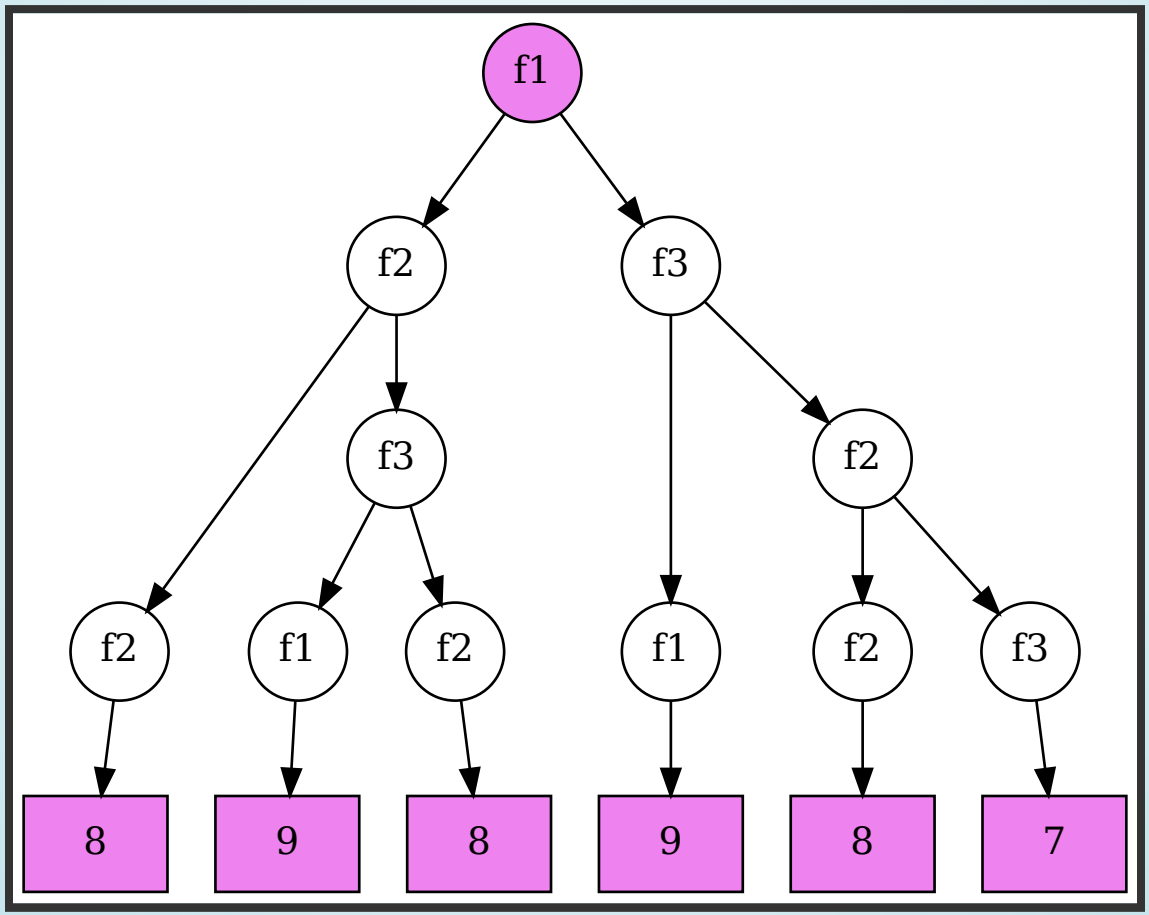
考察

関数はある区間の数列を出力する



考察

f_1 が全区間の数列を出力できるか分かればよい



DPテーブルの設計

i 番目の関数が，区間 $[l, r]$ の数列

$$x_l, \ldots, x_r$$

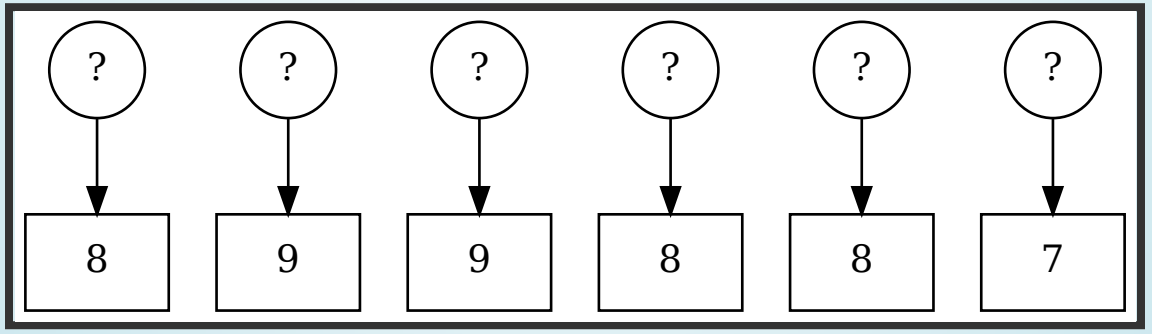
を出力できるかを `dp[l][r][i]` に保存

1. まず全要素を `false` にする
2. 出力可能と分かった (l, r, i) を `true` にしていく
3. `dp[0][N-1][0]` が `true` になったら答えは `Yes`
   - 区間 $[0, N-1]$ の数列を 0 番目の関数が生成可能

Q. まず求める区間は?

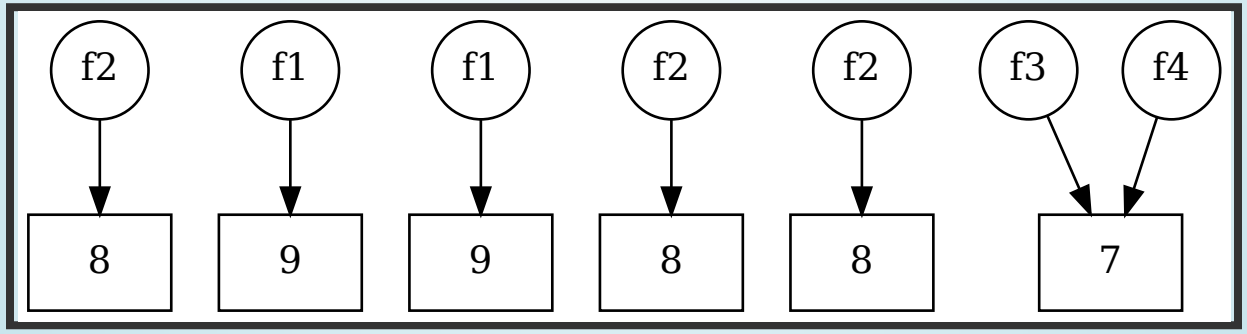
A. $l = r$ のとき.

- 区間 $[l, l]$ に対応する数列は x_l
- 1個の整数を出力するのは $f_i \to x$ の形の規則だけ
  - $f_i \to f_j\ f_k$ は必ず2個以上の整数を出力



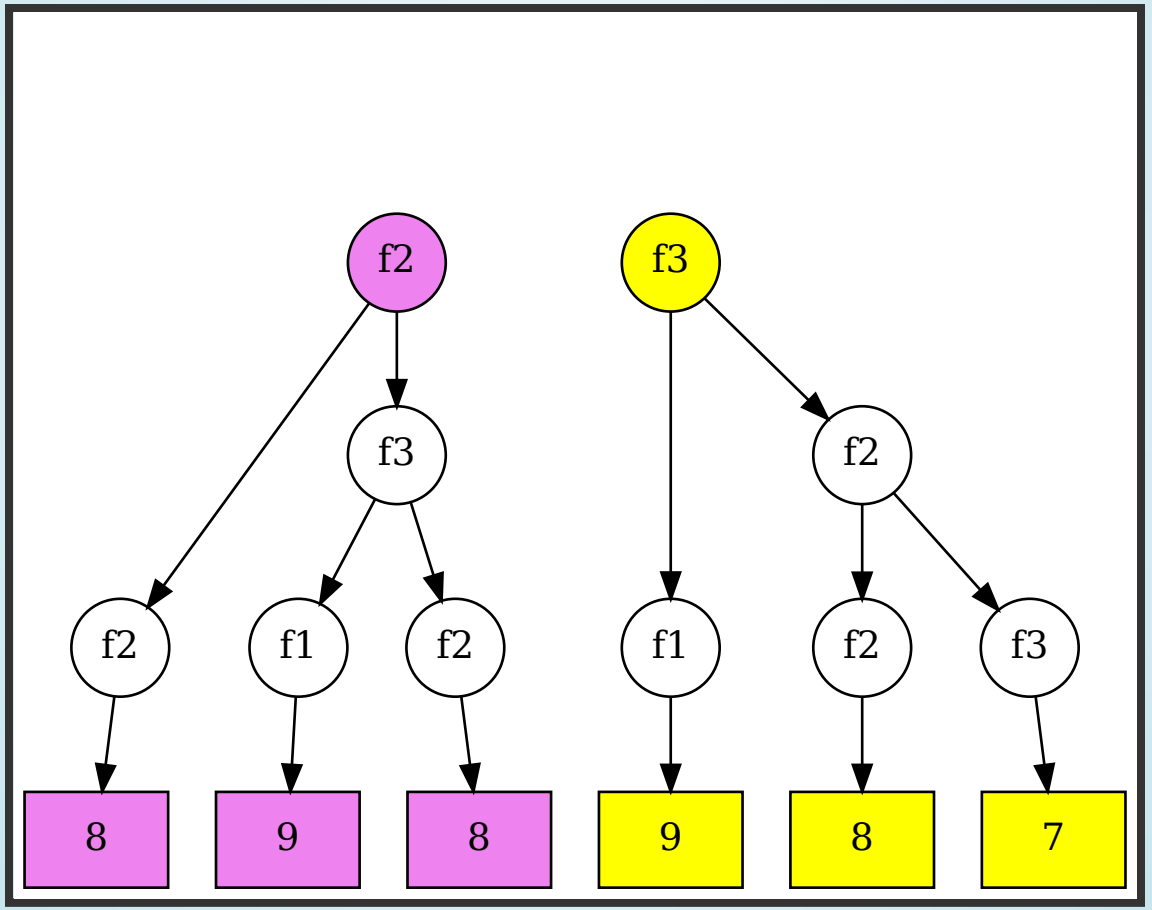
$l = r$ のとき

- $f_i \to x_l$ となる i を探す
  - 見つかったら `dp[l][l][i]` を `true` に
- 計算量は $\mathcal{O}(NM)$
  - $x_1, \ldots, x_N$ について f_i を探す
  - 各 x_l に対して M 個の関数を順に調べる
    - もちろん，複数見つかることもある



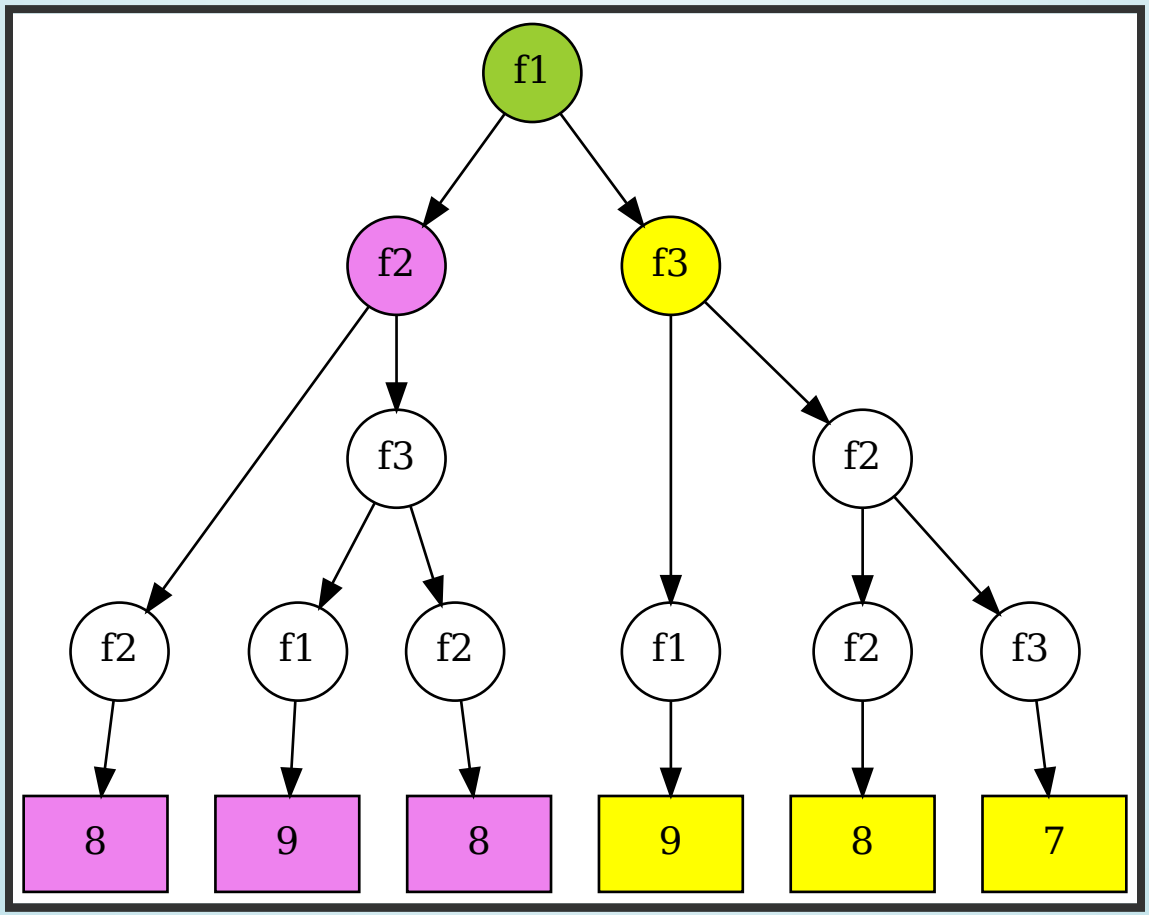
$l < r$ のとき

もし，$f_1 \to f_2\ f_3$ で，下の図のようになっていたら...



$l < r$ のとき

こうできるん!



$l < r$ のとき

$f_i \to f_j\ f_k$ であり,

- $l < p \leq r$ (数列のピンクと黄色の境界)
- `dp[l][p-1][j]` が `true` (ピンクの部分)
- `dp[p][r][k]` が `true` (黄色の部分)

となる p が存在すれば `dp[l][r][i] = true`

テーブルの更新順序

- つまり，隣接する2つの区間をマージしていく
- 幅 $(r - l + 1)$ のせまい区間から順に計算すればよい

```
for (int width = 2; width <= N; ++width) {
    for (int l = 0; l + width - 1 < N; ++l) {
        int r = l + width - 1;
        for (int i = 0; i < M; ++i) {
            for (int p = l + 1; p <= r; ++p) {
                int j = b[i], k = c[i];
                if (dp[l][p - 1][j] && dp[p][r][k]) {
                    dp[l][r][i] = true;
                    break;
                }
            }
        }
    }
}
```

$l < r$ の全体の計算量

$$\mathcal{O}(N^3 M)$$

- $l < r$ を満たす (l, r) は $\mathcal{O}(N^2)$ 通り
- (l, r) に対して M 個の関数を順に調べる
- (l, r, f_i) に対して調べる計算量は $\mathcal{O}(N)$
  - $l < p \leq r$ を満たす p の個数に依存

結論

- ボトムアップな DP で高速に求めることができる
  - $\mathcal{O}(NM) + \mathcal{O}(N^3M) = \mathcal{O}(N^3M)$
- 同じテーブルを使ったメモ化再帰でも大丈夫
  - メモ化再帰ならトップダウンでもできる

WRITER解

- 青木 Java 43行
- 井上 C++ 43行
- 鈴木 C++ 35行
- 田中 C++ 59行, Python 31行

提出状況

- First Accept
  - onsite: yakzto さん (1:08)
  - online: natsugiri さん (0:14)
- 正答率 18/27 (67 %)