ハフマン·ミーリーの最小化法（k-等価性分類法）

という名前がついているらしい.
この名前のソースは http://www.amazon.co.jp/dp/4627805527/ の p.36.
要するに緑本に載っている最小化の方法です.

でやってみる

篠原　歩

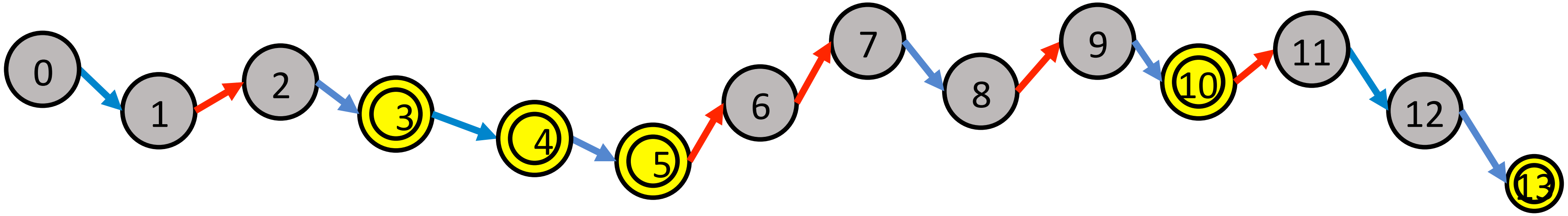
2014年1月13日

基本アイディア

- 緑本の最小化アルゴリズム
  状態の集合を，必要に駆られたときのみ
  分割していく.
- 「必要に駆られる」証拠があるため，
  最小性が保証できる.

- 通常のDFAが入力のときは，どの入力記号に対しても
  ちょうど行き先が1つ決まっているが，
  我々の場合には行き先が定義されていない場合があ
  るところが異なる.
- このことに注意しながら，なるべく分割を起こさない
  割り当てを考えていく.

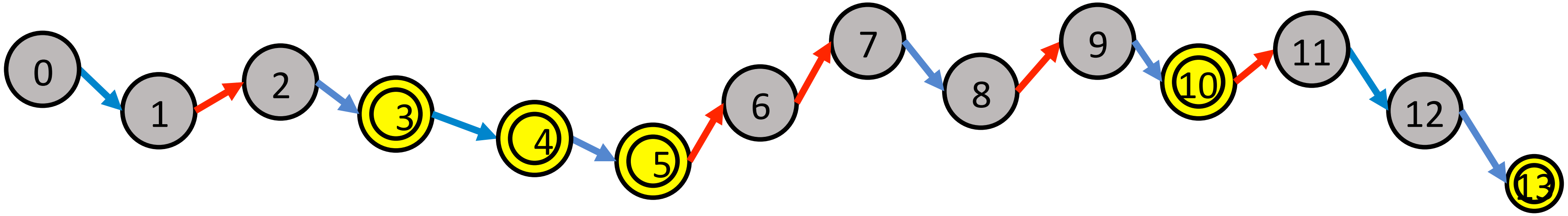
入力例 (by 上埜)



第一ステップとして，受理状態と非受理状態は当然ながら分割する必要がある.

$$\{0, 1, 2, 6, 7, 8, 9, 11, 12\} \quad \{3, 4, 5, 10, 13\}$$

入力例 (by 上埜)



第一ステップとして，受理状態と非受理状態は当然ながら分割する必要がある.

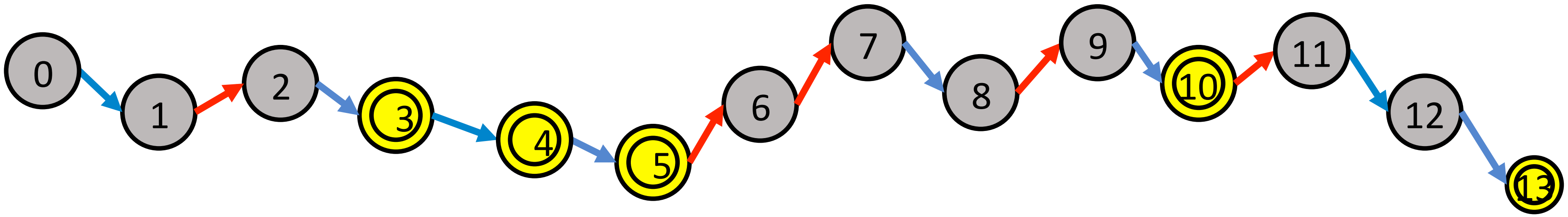
グループAと呼ぶ

$$\{0, 1, 2, 6, 7, 8, 9, 11, 12\}$$

グループBと呼ぶ

$$\{3, 4, 5, 10, 13\}$$

入力例 (by 上埜)



グループAと呼ぶ

{0, 1, 2, 6, 7, 8, 9, 11, 12}

- A - A - A - - -

グループBと呼ぶ

{3, 4, 5, 10, 13}

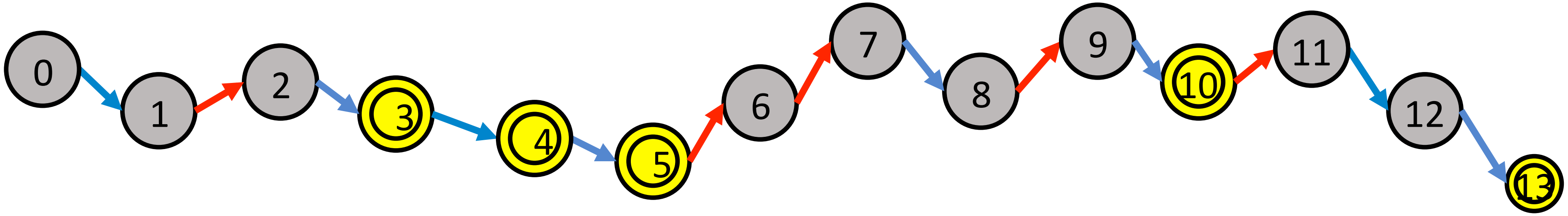
- - A A -

a遷移の行き先グループを求める.

注：行き先がないところは，- と書いている.

どちらのグループも，異なるグループには行かないので
この段階では分割の必要なし.

入力例 (by 上埜)



グループAと呼ぶ

$\{0, 1, 2, 6, 7, 8, 9, 11, 12\}$

A - B - A - B A B

グループBと呼ぶ

$\{3, 4, 5, 10, 13\}$

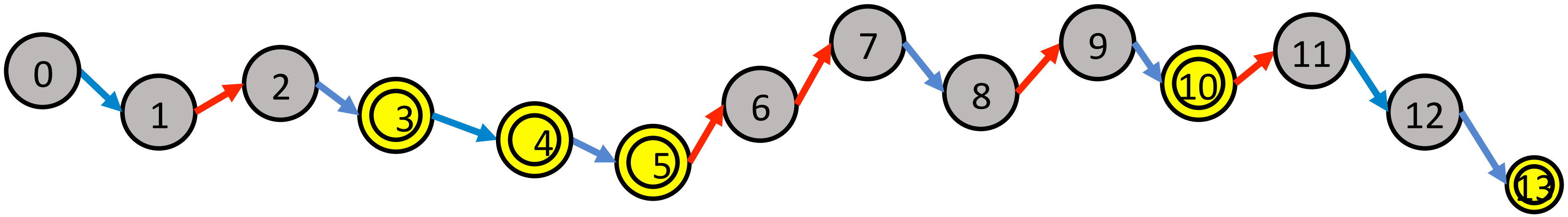
B B - - -

b遷移の行き先グループを求める.

グループBは 行き先が食い違っていないので
この段階では分割の必要はない.

一方, グループAは 行き先が食い違っているので,
必ず分割が必要となる.

入力例 (by 上埜)



グループAと呼ぶ

$\{0, 1, 2, 6, 7, 8, 9, 11, 12\}$

A - B - A - B A B

グループBと呼ぶ

$\{3, 4, 5, 10, 13\}$

B B - - -

Aにいくもの

$\{0, 7, 11\}$

グループCと呼ぶ

Bにいくもの

$\{2, 9, 12\}$

グループDと呼ぶ

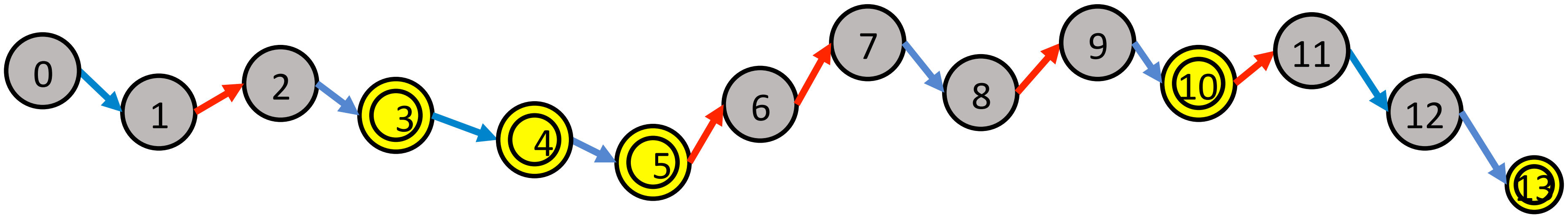
1, 6, 8

それぞれグループCまたはDに属する予定

属してもよい（どちらかに決める根拠がない）
もちろん，新たなグループにしてもよいが，そうすると
無駄に状態が増えるのでそれは避ける.

一方，グループAは 行き先が食い違っているので，
必ず分割が必要となる.

入力例 (by 上埜)

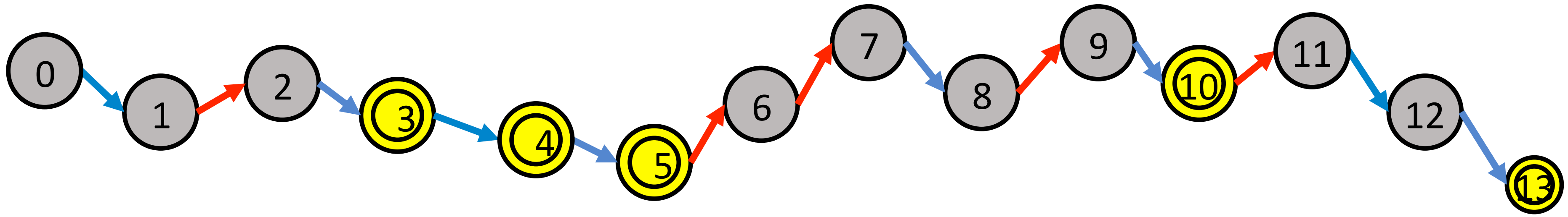


グループC	グループD	それぞれ CまたはD	グループB
{0, 7, 11}	{2, 9,12}	1, 6, 8	{3, 4, 5, 10, 13}
- - -	- - -	D C D	- - C or D C -

a遷移の行き先グループを求める.

グループB において,
5の行き先である6 は “C or D” だが
Dにすると必ず分割する必要があるため
C を選んだ方がよさそう.

入力例 (by 上埜)



グループC	グループD	それぞれ CまたはD	グループB
{0, 7, 11}	{2, 9,12}	1, 6, 8	{3, 4, 5, 10, 13}
- - -	- - -	D C D	- - C or D C -
C or D, C or D, D	B B B	- - -	B B - - -

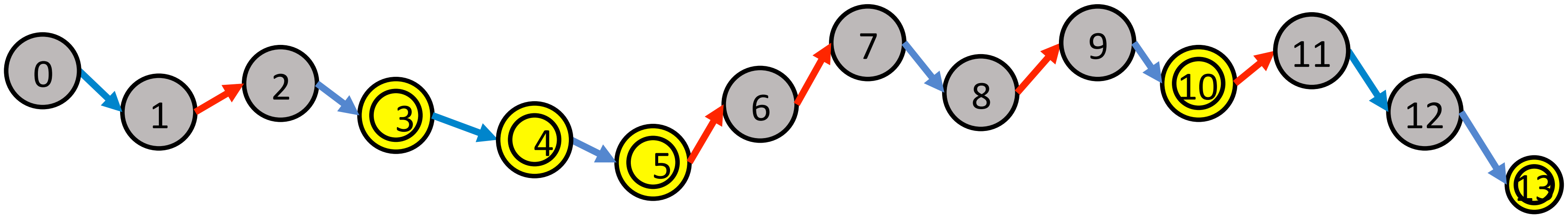
a遷移の行き先グループを求める.

b遷移の行き先グループを求める.

グループC において,
0 と 7 の行き先である 1 と 8 は
どちらも D を選んだ方がよさそうだ.

グループB において,
5の行き先である6 は “C or D” だが
Dにすると必ず分割する必要があるため
C を選んだ方がよさそう.

入力例 (by 上埜)



この観察に基づいて代入してみる.

グループC	グループD	D	C	D	グループB
{0, 7, 11}	{2, 9,12}	1,	6,	8	{3, 4, 5, 10, 13}
- - -	- - -	D	C	D	- - C or D C -
C or D C or D D	B B B	-	-	-	B B - - -

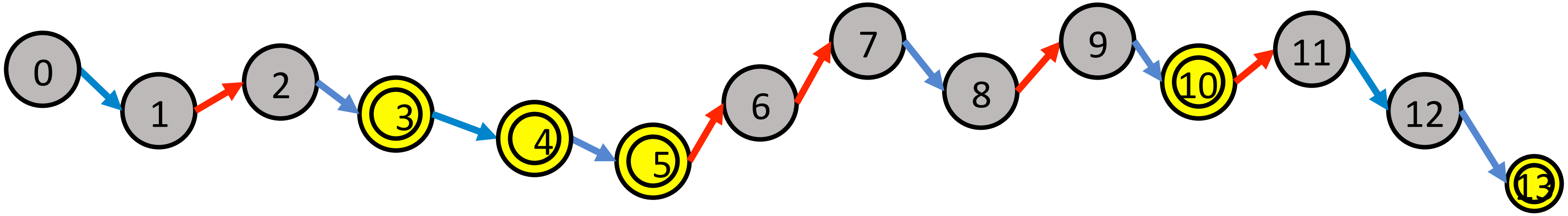
a遷移の行き先グループを求める.

b遷移の行き先グループを求める.

グループC において,
0 と 7 の行き先である 1 と 8 は
どちらも D を選んだ方がよさそうだ.

グループB において,
5の行き先である6 は “C or D” だが
Dにすると必ず分割する必要があるため
C を選んだ方がよさそう.

入力例 (by 上埜)



この観察に基づいて代入してみる.

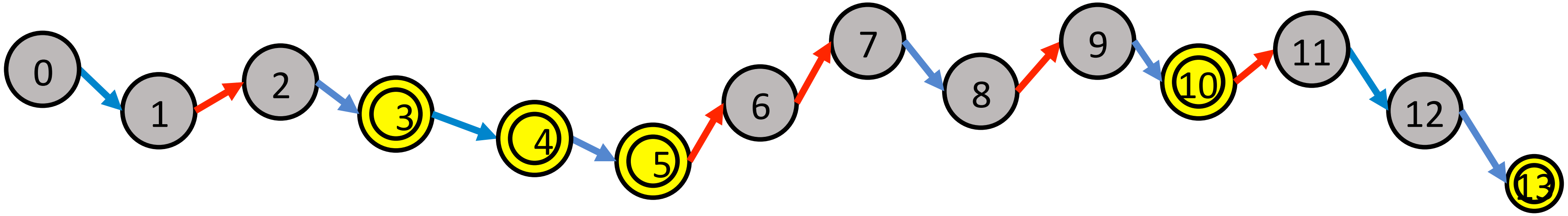
グループC	グループD	D	C	D	グループB
{0, 7, 11}	{2, 9,12}	1,	6,	8	{3, 4, 5, 10, 13}
- - -	- - -	D	C	D	- - C C -
D D D	B B B	-	-	-	B B - - -

a遷移の行き先グループを求める.

b遷移の行き先グループを求める.

行き先を更新してみると,
確かに辻褄が合っており,
これ以上, 分割の必要もない.

入力例 (by 上埜)



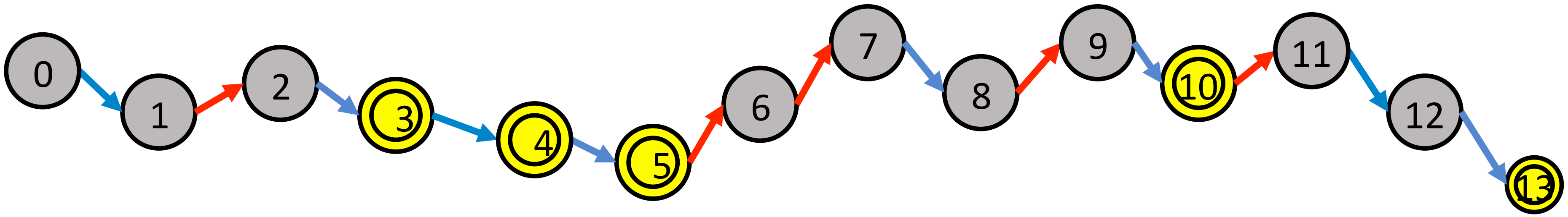
この観察に基づいて代入してみる.

グループC	グループD	D	C	D	グループB
{0, 7, 11}	{2, 9,12}	1,	6,	8	{3, 4, 5, 10, 13}
- - -	- - -	D	C	D	- - C C -
D D D	B B B	-	-	-	B B - - -

分かり易いように並べ替えておく

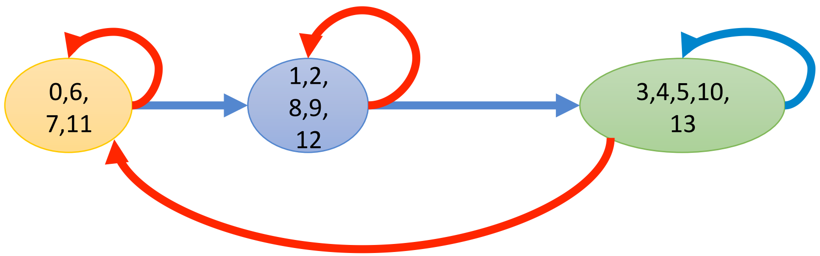
グループC	グループD	グループB
{0, 6, 7, 11}	{1, 2, 8, 9,12}	{3, 4, 5, 10, 13}
- C - -	D - D - -	- - C C -
D - D D	- B - B B	B B - - -

入力例 (by 上埜)



確かにこれで辻褄が合ったので，これ以上分割を試す必要がない(緑本に載っている証明の通り).

このことにより，この分割が最小であることも保証されている.



分かり易いように並べ替えておく

グループC

{0, 6, 7, 11}

- C - -

D - D D

グループD

{1, 2, 8, 9,12}

D - D - -

- B - B B

グループB

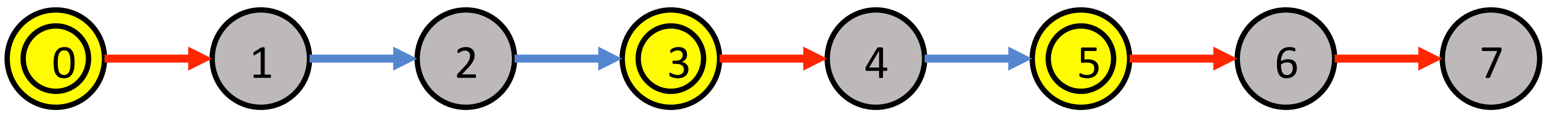
{3, 4, 5, 10, 13}

- - C C -

B B - - -

ここまでのコメント

- この方法で常に簡単に求められるかどうかがまだ確信が持てていないが，最小性を保証しやすいアプローチだと思う.
- 下薗先生の作った入力例



でも手計算で試してみましたがうまくいきました
（最後の状態の場所についての別解も求まりました）

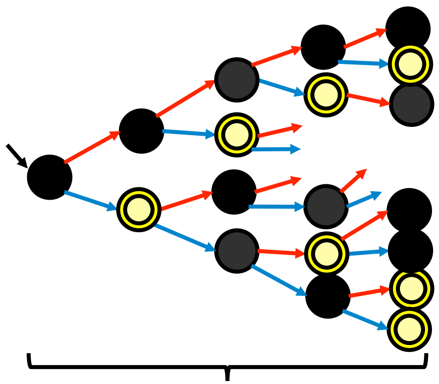
さらなる今後の展望

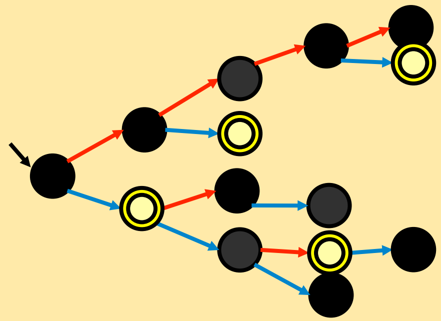
もしもこのアプローチで多項式時間で
最小化ができるとすると，prefix sample ではなくても，“prefix closedなサンプル” に対してそのまま拡張できそうな期待が持てる．

定義：
例の集合 $S = Pos \cup Neg$ が prefix closed であるとは，
任意の $w \in S$ に対して w のprefix がすべて S に含まれている
ときをいう（ことにしましょう）．

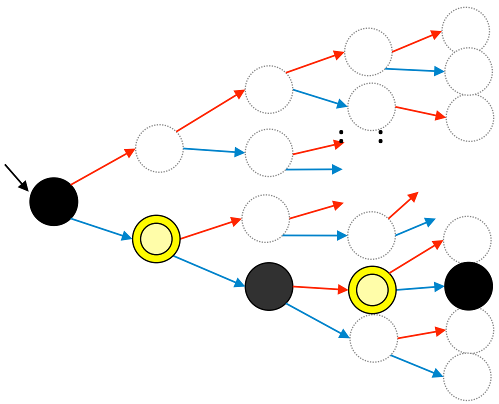
つまりパスの途中には穴がない

下記の3つのサンプルは，どれも prefix closed.
つまり，完全サンプルと 完全prefix sample を合わせて拡張した概念のつもり．



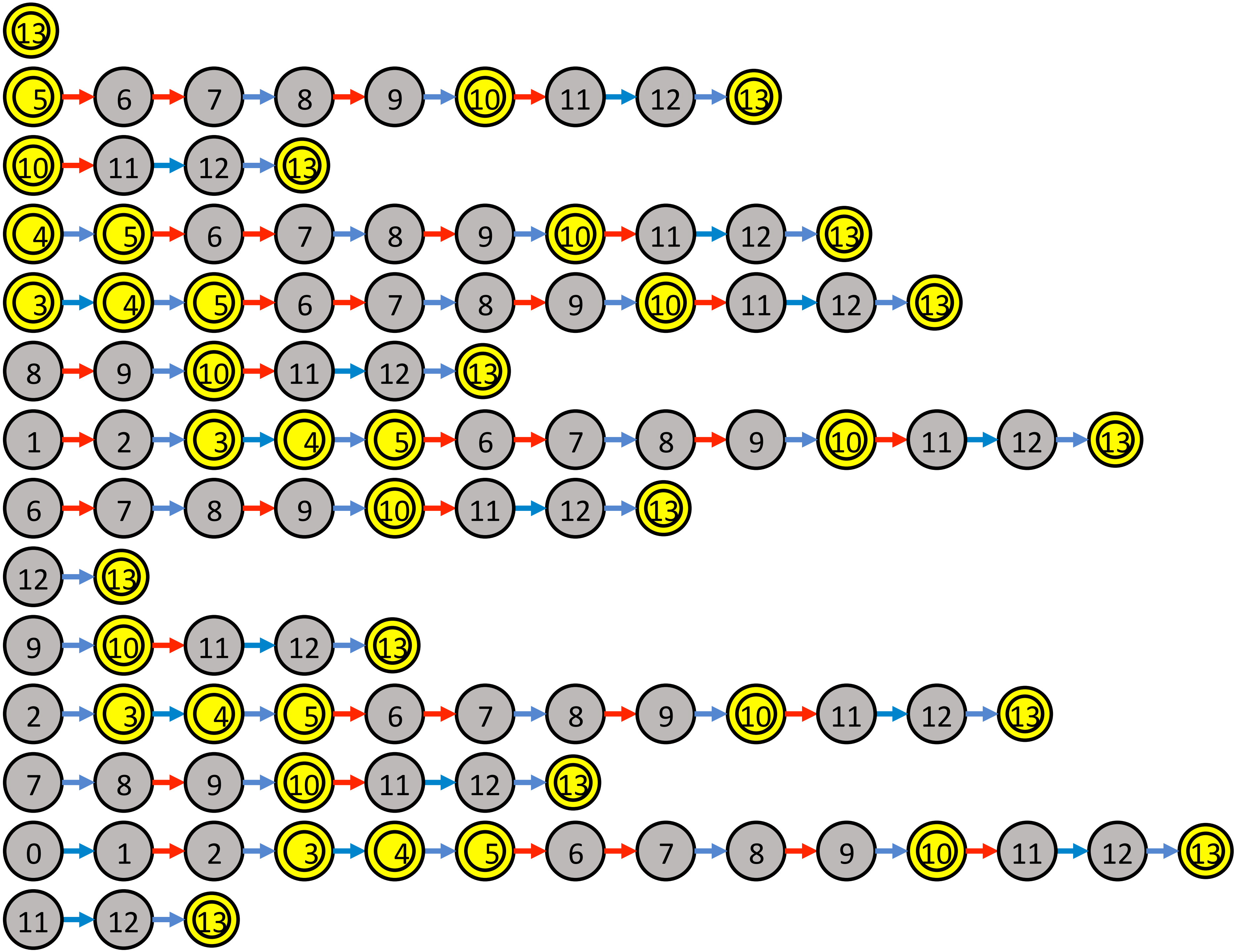


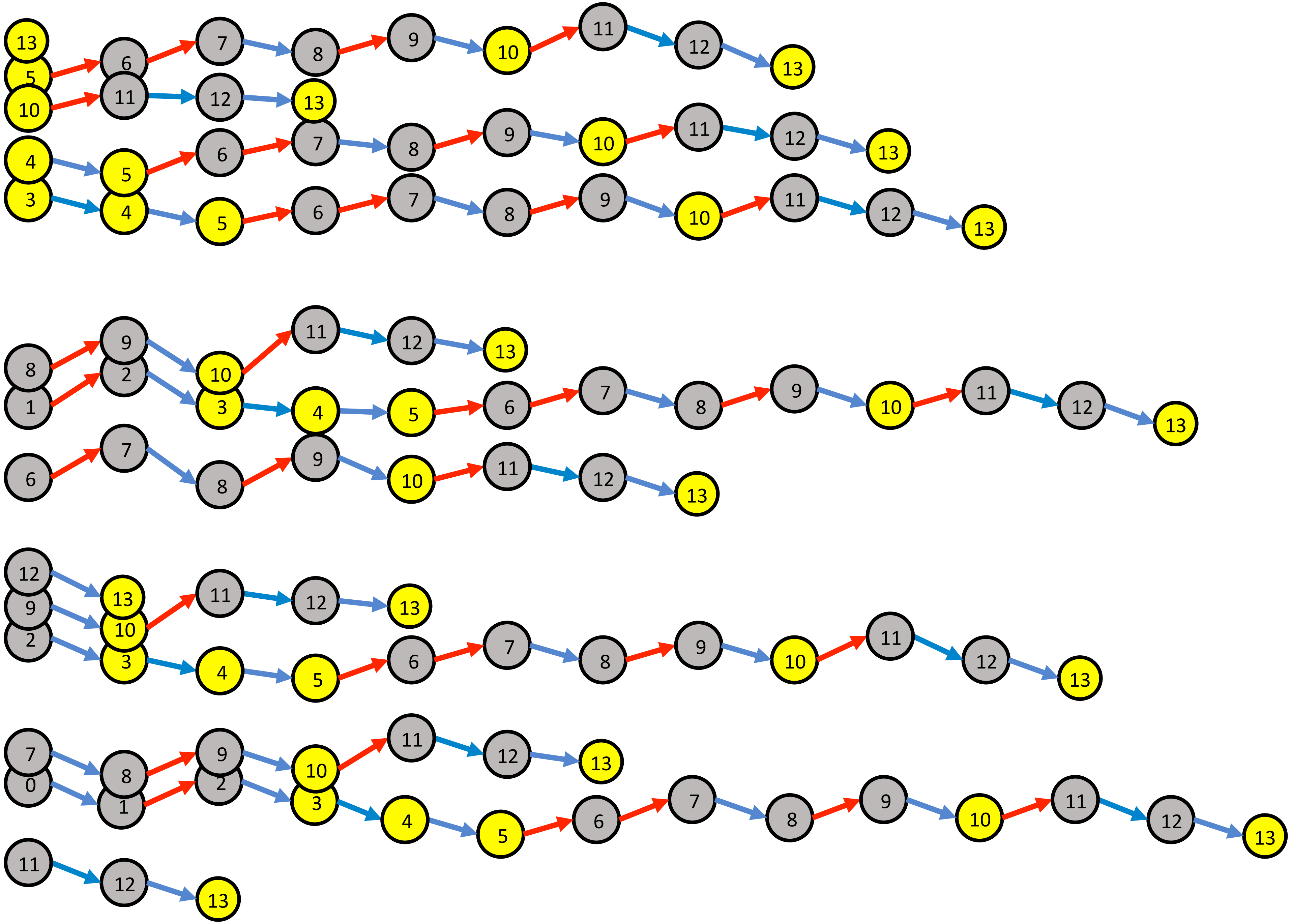
x^{ε}不完全であっても，
prefix closed ならば P ?

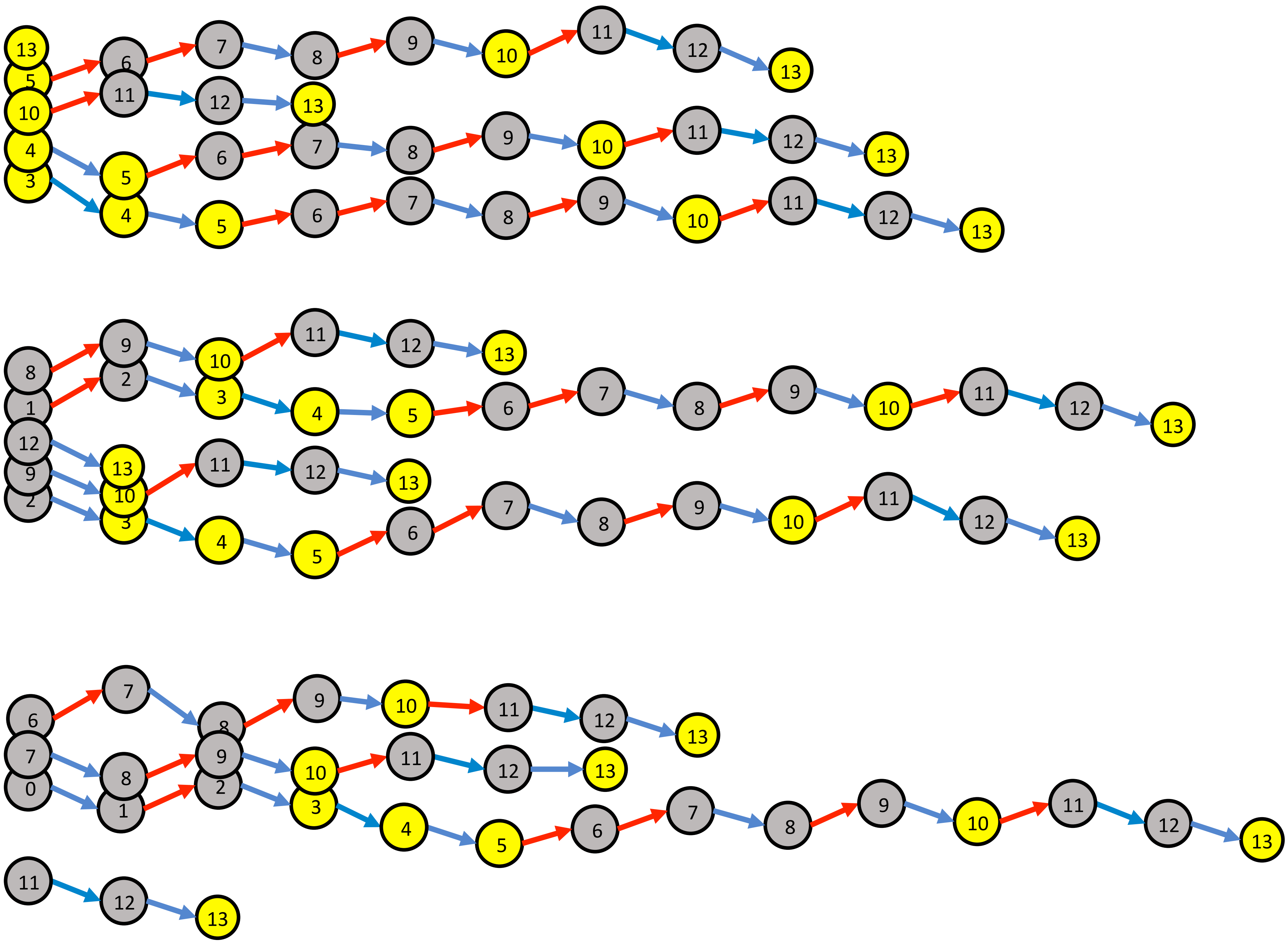


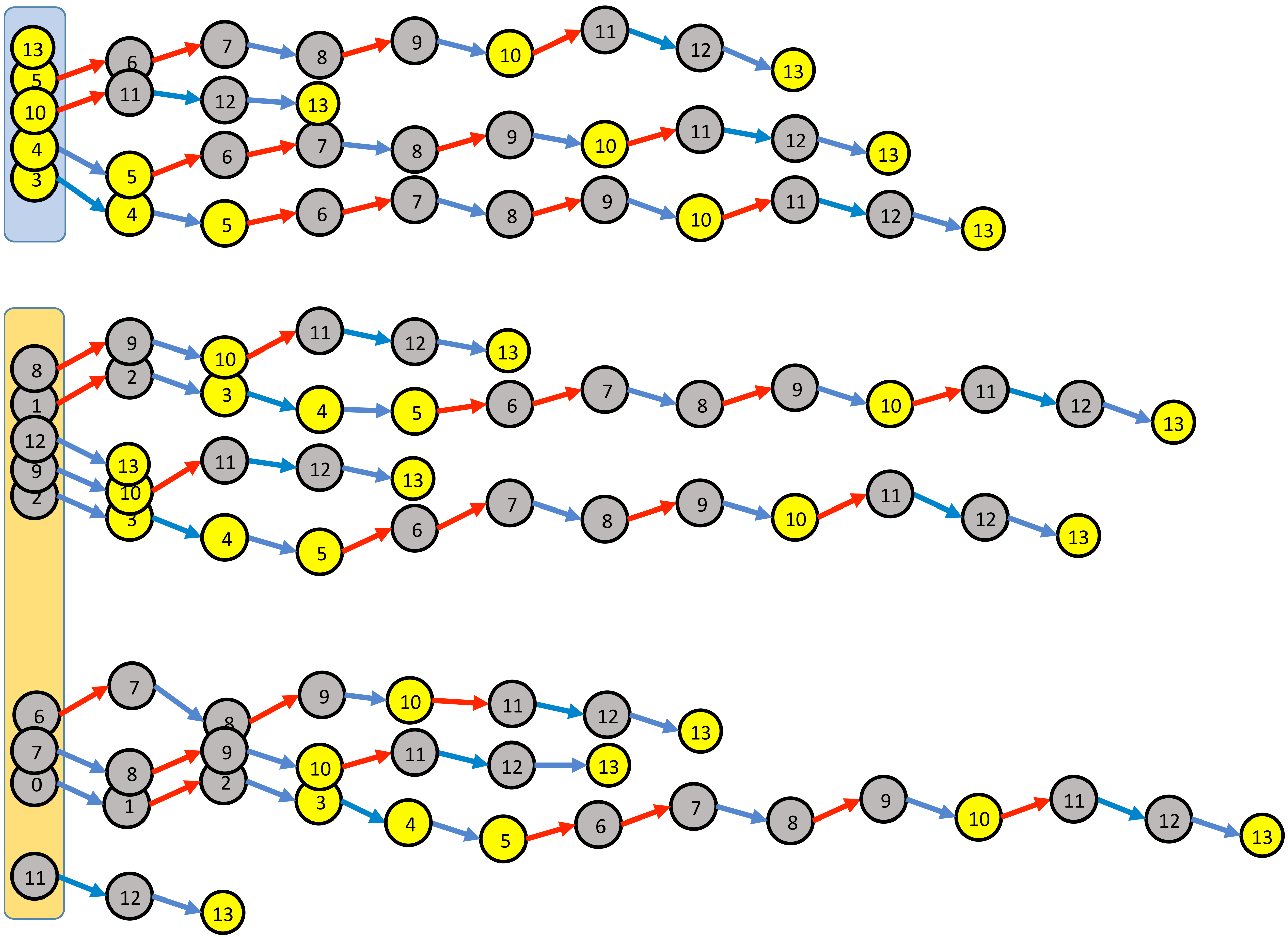
完全prefix sample

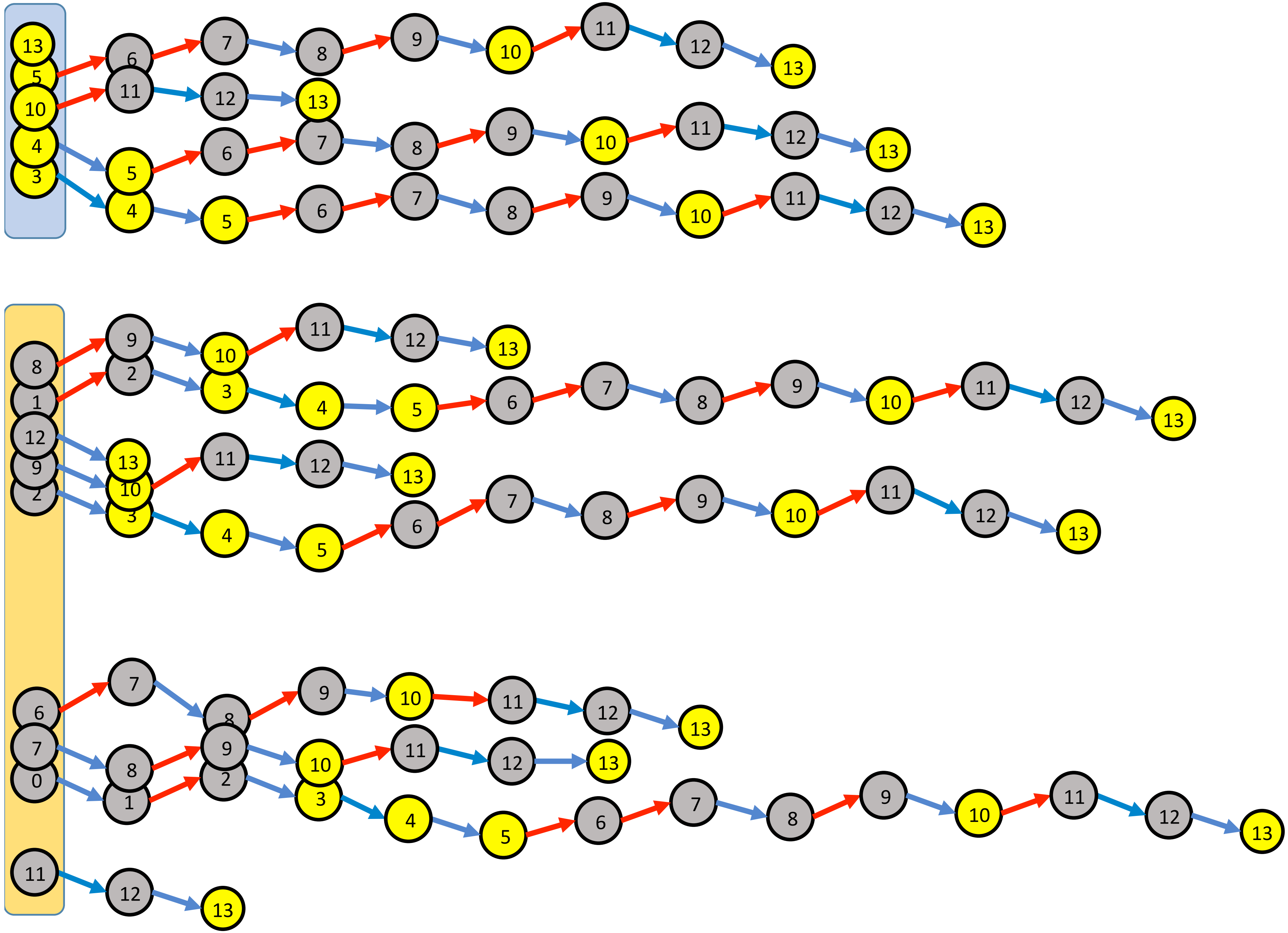
以下，作業の残骸

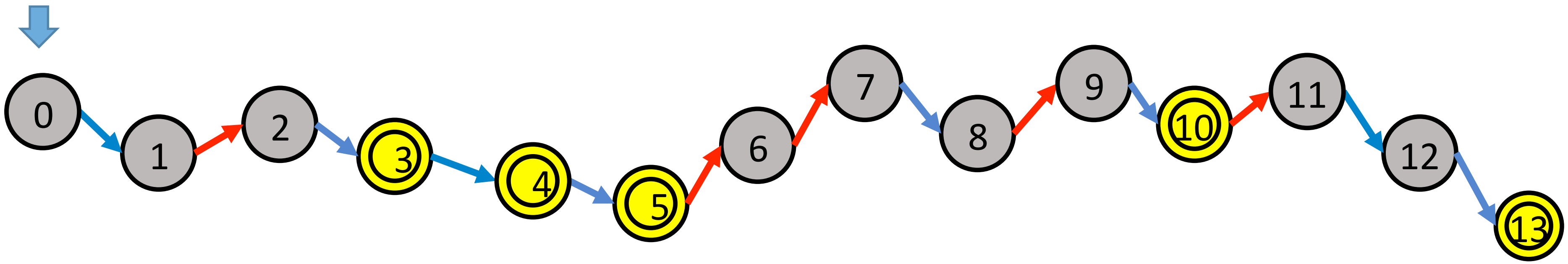

13
5 6 7 8 9 10 11 12 13
10 11 12 13
4 5 6 7 8 9 10 11 12 13
3 4 5 6 7 8 9 10 11 12 13
8 9 10 11 12 13
1 2 3 4 5 6 7 8 9 10 11 12 13
6 7 8 9 10 11 12 13
12 13
9 10 11 12 13
2 3 4 5 6 7 8 9 10 11 12 13
7 8 9 10 11 12 13
0 1 2 3 4 5 6 7 8 9 10 11 12 13
11 12 13



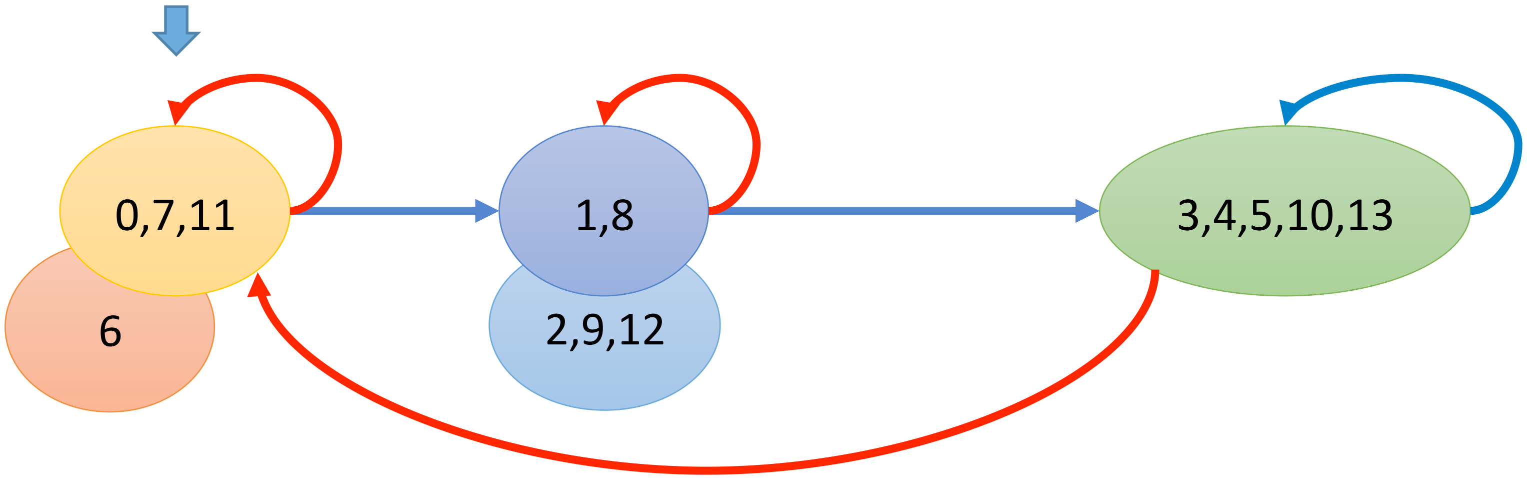

13
5 6 7 8 9 10 11 12 13
10 11 12 13
4 5 6 7 8 9 10 11 12 13
3 4 5 6 7 8 9 10 11 12 13
8 9 10 11 12 13
1 2 3 4 5 6 7 8 9 10 11 12 13
12 13
9 10 11 12 13
2 3 4 5 6 7 8 9 10 11 12 13
6 7 8 9 10 11 12 13
7 8 9 10 11 12 13
0 1 2 3 4 5 6 7 8 9 10 11 12 13
11 12 13







0
1
2
3
4
5
6
7
8
9
10
11
12
13



0,7,11
6
1,8
2,9,12
3,4,5,10,13