Suffix Automaton を完全に理解する

JOI 夏季セミナー 2020　KoD

はじめに

- 高速文字列解析班の KoD です
- Suffix Automaton について紹介します！

はじめに

- Suffix Automaton って，何ができるの…?

はじめに

- Suffix Automaton って，何ができるの…?
  - 部分文字列の種類数を数える

はじめに

- Suffix Automaton って，何ができるの…?
  - 部分文字列の種類数を数える
  - 部分文字列を辞書順に並べたときの k 番目を求める

はじめに

- Suffix Automaton って，何ができるの…?
  - 部分文字列の種類数を数える
  - 部分文字列を辞書順に並べたときの k 番目を求める
  - 他にもたくさん（後ほど紹介）

はじめに

- Suffix Automaton って，何ができるの…?
  - 部分文字列の種類数を数える
  - 部分文字列を辞書順に並べたときの k 番目を求める
  - 他にもたくさん（後ほど紹介）

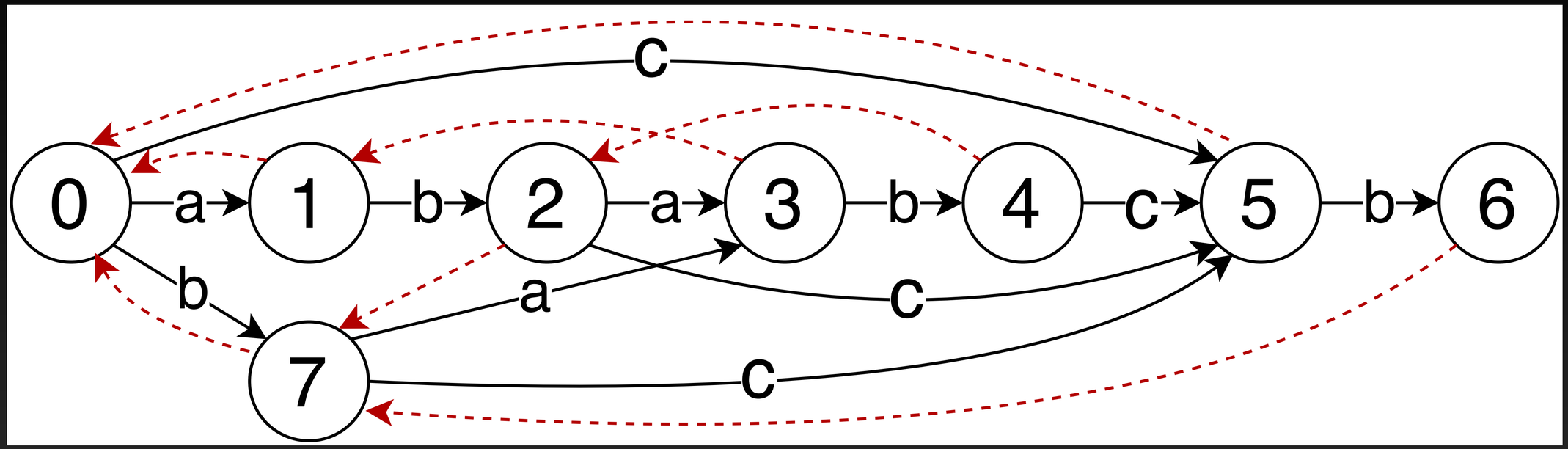
つよそう！（小並感）

目次

1. Suffix Automaton の性質
2. 具体的な構築方法
3. 計算量などの解析
4. 実際に使ってみよう！

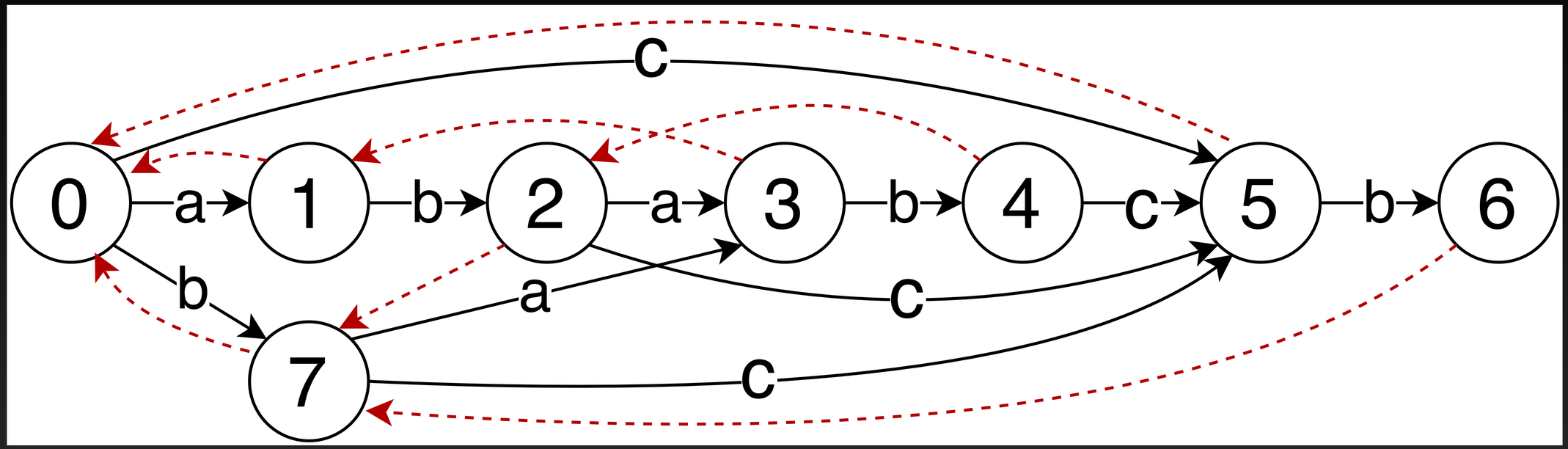
- ▶ 1. Suffix Automaton の性質
- 2. 具体的な構築方法
- 3. 計算量などの解析
- 4. 実際に使ってみよう！

Suffix Automaton の性質



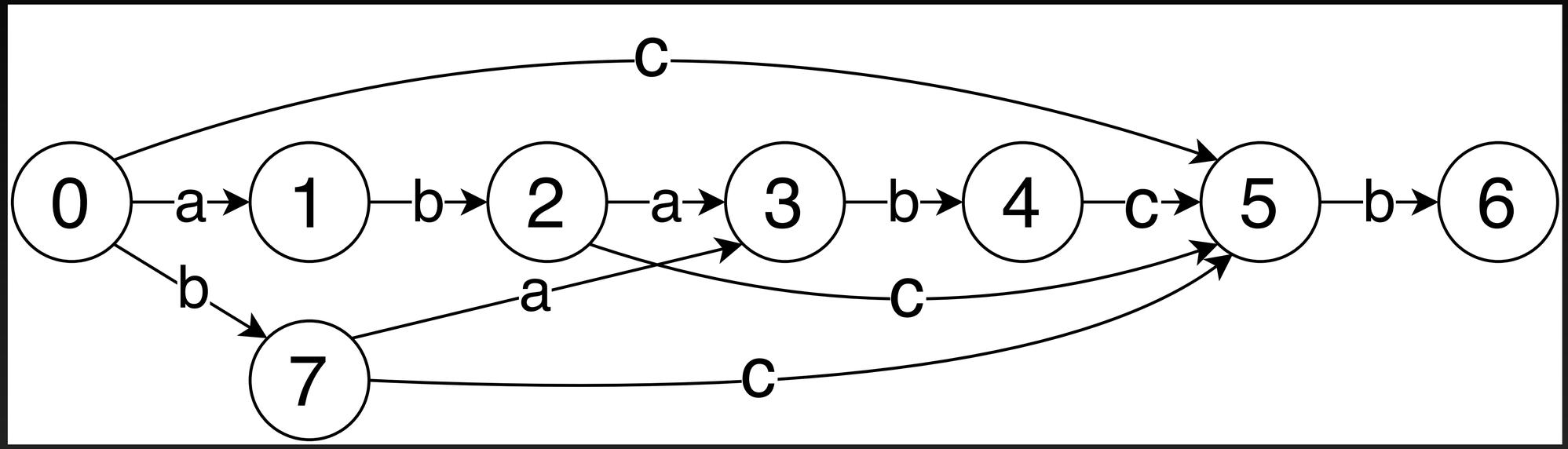
- 文字列 S（上の例では “ababcb”）
  の部分文字列を，漏れなく重複なく管理

Suffix Automaton の性質



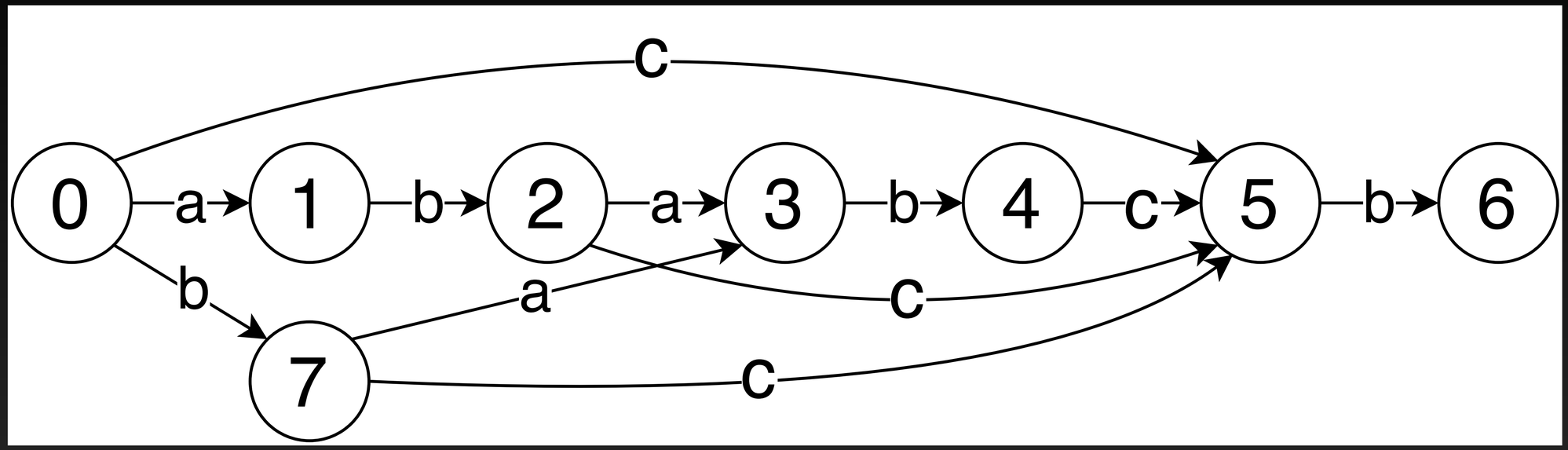
- 図が複雑すぎる
  -> 実線と破線を分離

Suffix Automaton の性質



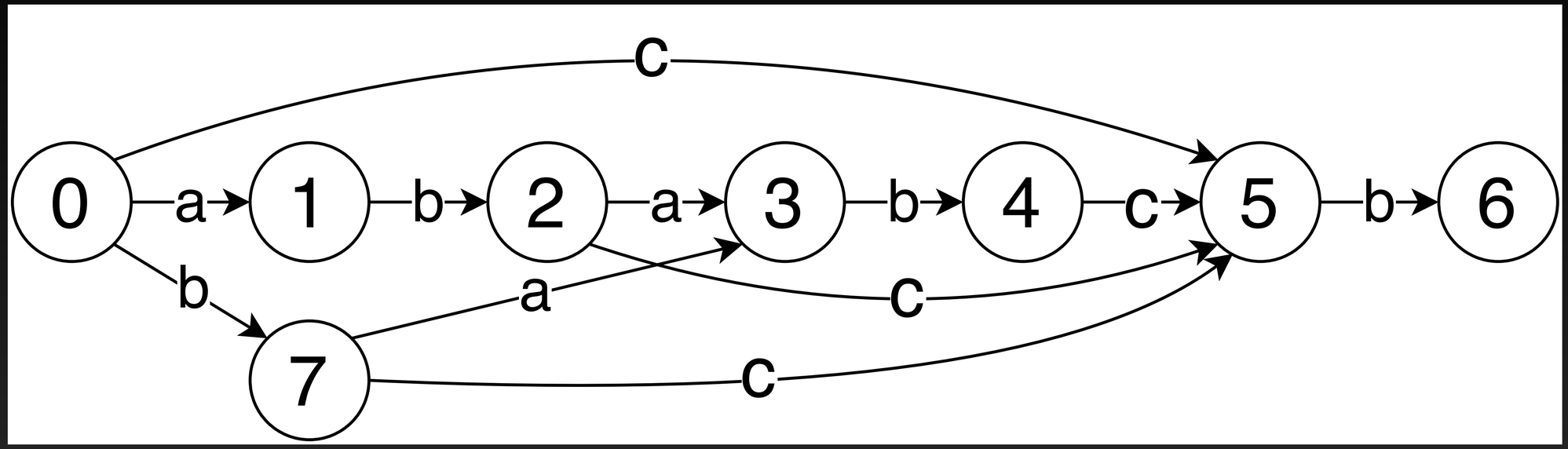
- 実線部分だけ取り出すと，DAG になっている

Suffix Automaton の性質



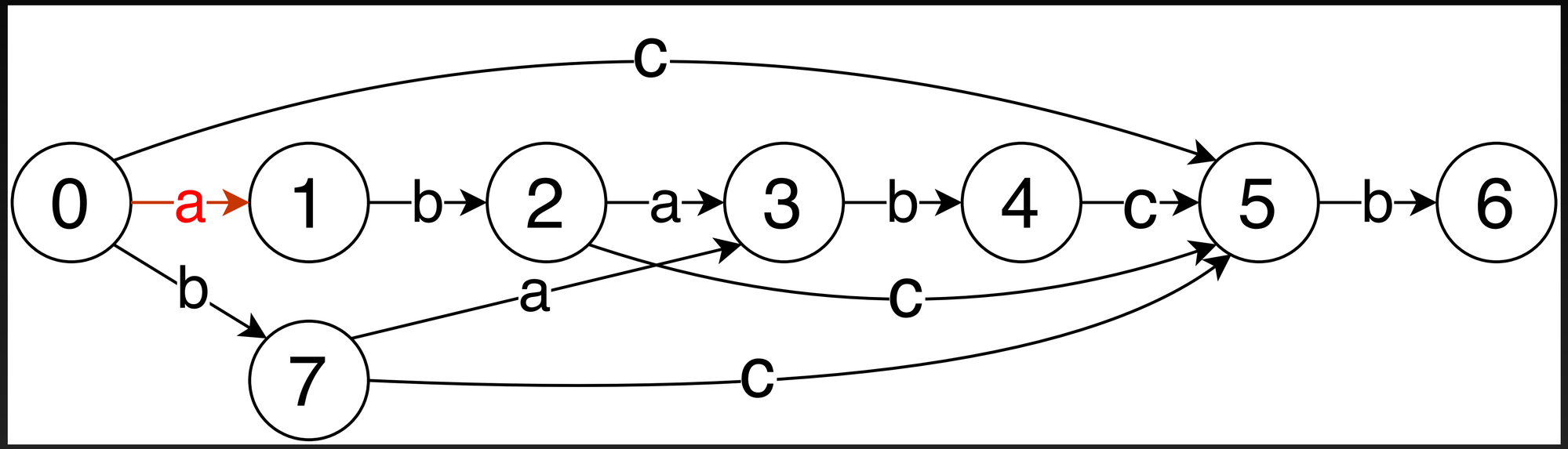
- 各ノードは，始点からのパスで表されるような部分文字列を管理

Suffix Automaton の性質



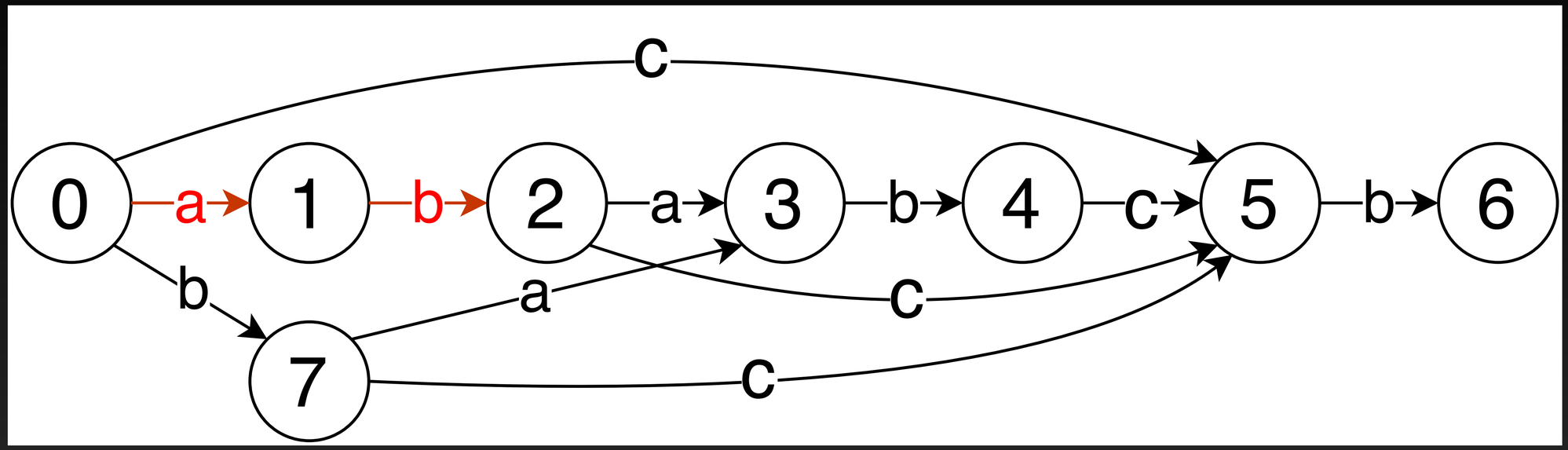
ノード	1	2	3	4	5	6	7
管理する 部分文字列							

Suffix Automaton の性質



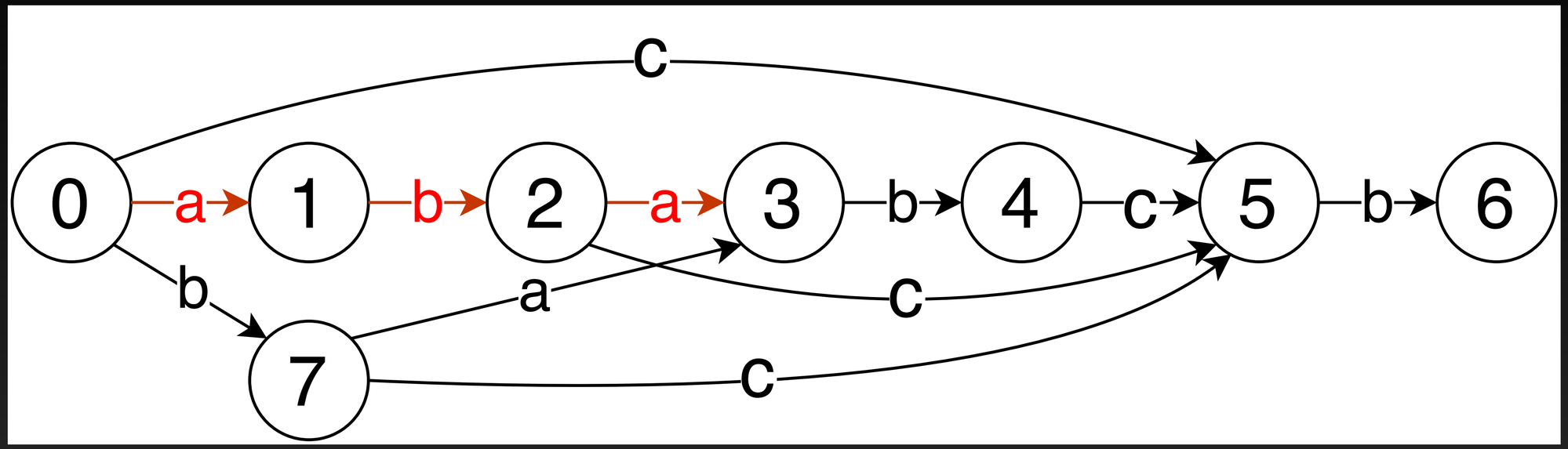
ノード	1	2	3	4	5	6	7
管理する 部分文字列	a						

Suffix Automaton の性質



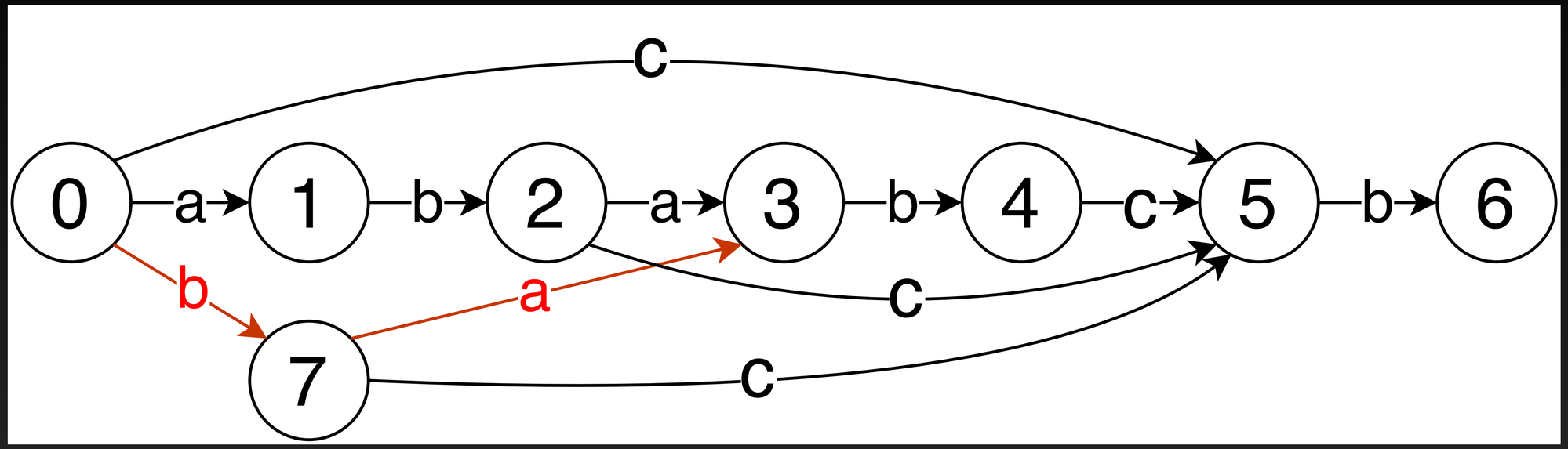
ノード	1	2	3	4	5	6	7
管理する 部分文字列	a	ab					

Suffix Automaton の性質



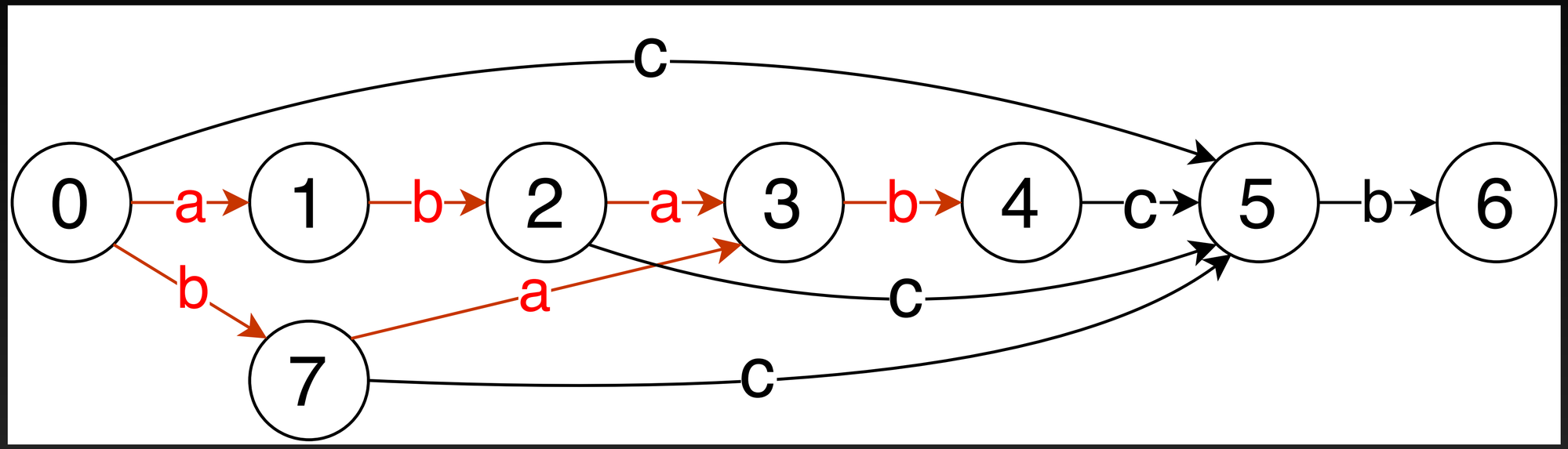
ノード	1	2	3	4	5	6	7
管理する 部分文字列	a	ab	aba				

Suffix Automaton の性質



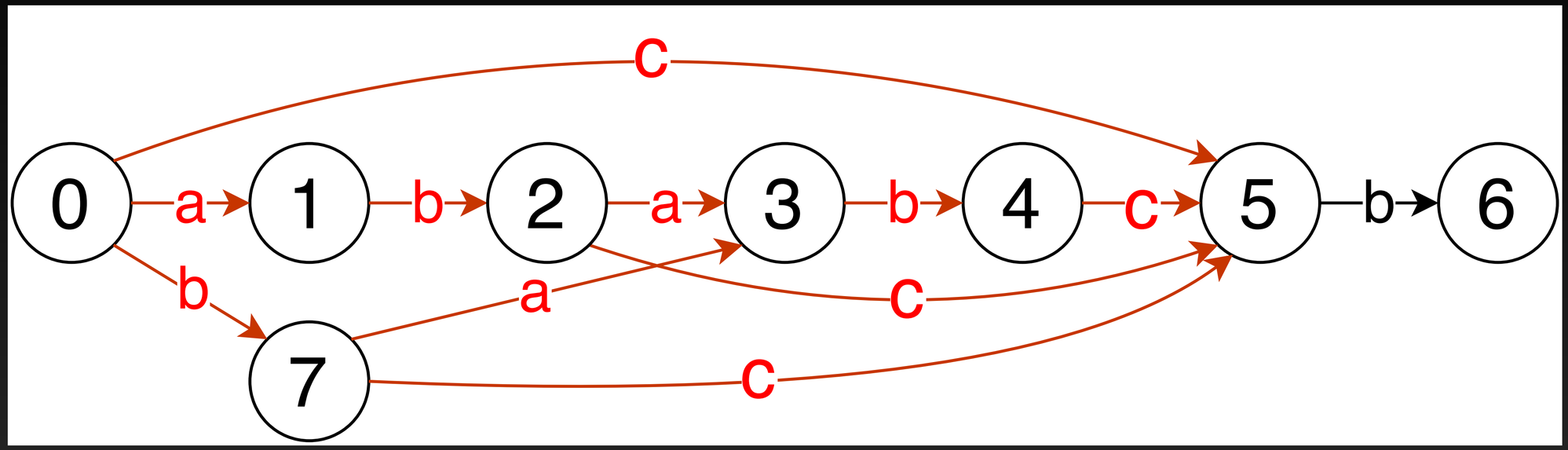
ノード	1	2	3	4	5	6	7
管理する 部分文字列	a	ab	aba ba				

Suffix Automaton の性質



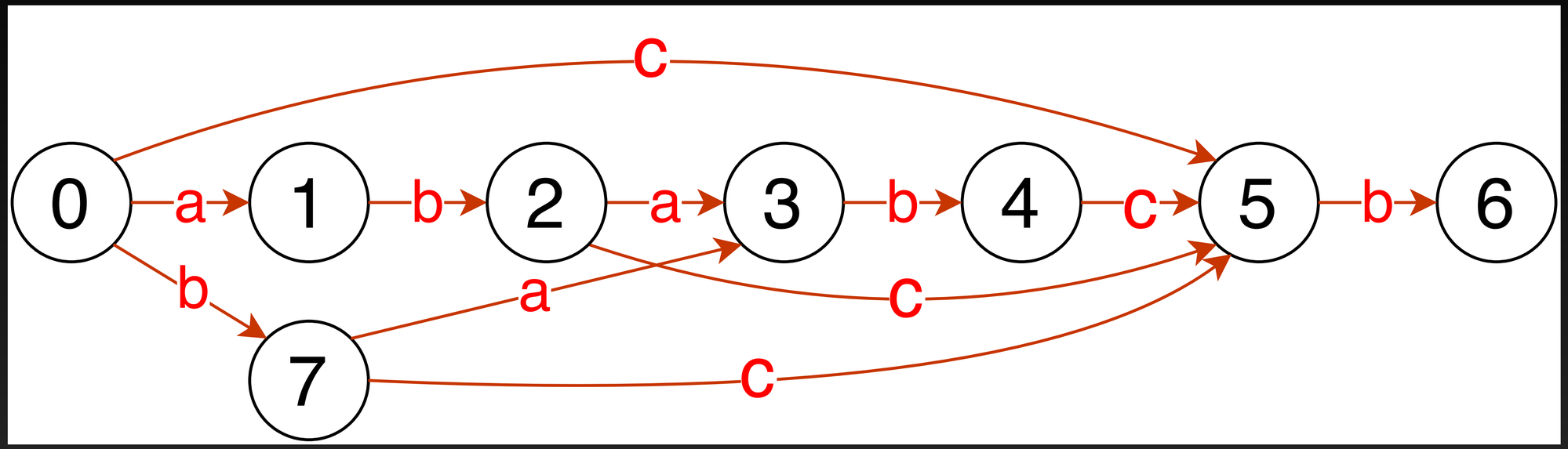
ノード	1	2	3	4	5	6	7
管理する 部分文字列	a	ab	aba ba	abab bab			

Suffix Automaton の性質



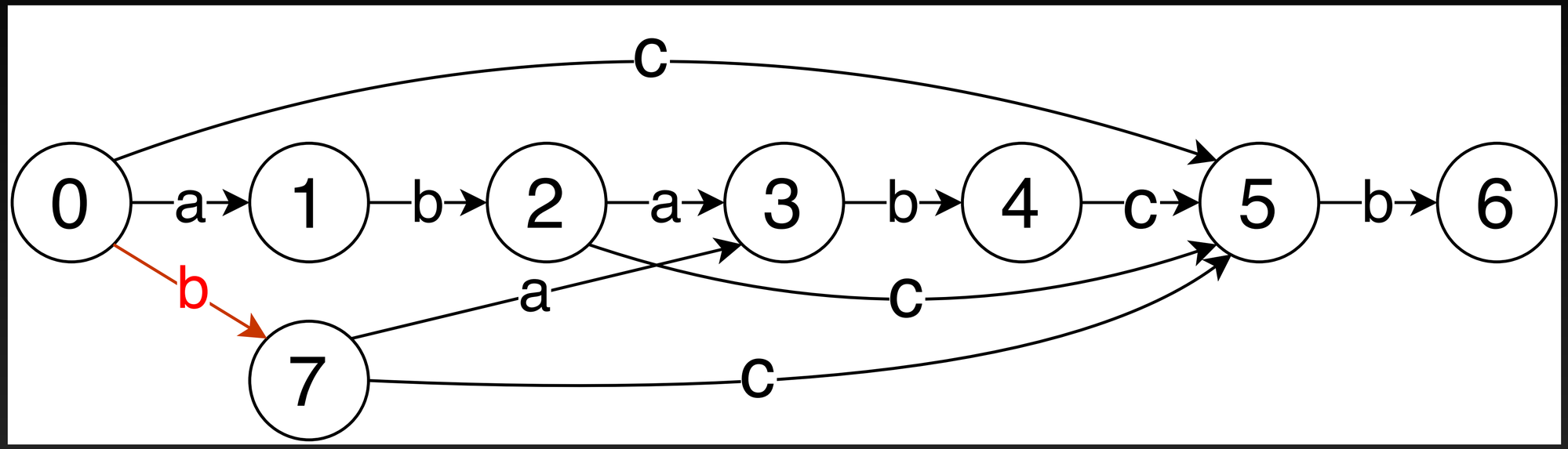
ノード	1	2	3	4	5	6	7
管理する 部分文字列	a	ab	aba ba	abab bab	ababc babc abc bc c		

Suffix Automaton の性質



ノード	1	2	3	4	5	6	7
管理する 部分文字列	a	ab	aba ba	abab bab	ababc babc abc bc c	ababcb babcb abcb bob cb	

Suffix Automaton の性質



ノード	1	2	3	4	5	6	7
管理する 部分文字列	a	ab	aba ba	abab bab	ababc babc abc bc c	ababcb babcb abcb bob cb	b

Suffix Automaton の性質

ノード	1	2	3	4	5	6	7
管理する 部分文字列	a	ab	aba ba	abab bab	ababc babc abc bc c	ababcb babcb abcb bob cb	b

- 重要な性質: 各ノードが管理する文字列は，順番に先頭を削っていったものになっている

Suffix Automaton の性質

ノード	1	2	3	4	5	6	7
管理する 部分文字列	a	ab	aba ba	abab bab	ababc babc abc bc c	ababcb babcb abcb bob cb	b

- つまり，ノード u が表す最長の文字列を T[u] で表すと，ノード u が管理する文字列は T[u] の接尾辞

Suffix Automaton の性質

ノード	1	2	3	4	5	6	7
管理する 部分文字列	a	ab	aba ba	abab bab	ababc babc abc bc c	ababcb babcb abcb bob cb	b

- 前から削っていき，途中で無くなる場合がある（ノード 4 など）

Suffix Automaton の性質

ノード	1	2	3	4	5	6	7
管理する 部分文字列	a	ab	aba ba	abab bab ab b	ababc babc abc bc c	ababcb babcb abcb bob cb	b

- 前から削っていき，途中で無くなる場合がある（ノード 4 など）

Suffix Automaton の性質

ノード	1	2	3	4	5	6	7
管理する 部分文字列	a	ab	aba ba	abab bab ab b	ababc babc abc bc c	ababcb babcb abcb bob cb	b

- この場合，“ab” はノード 2 が管理する最も長い文字列（すなわち，T[2]）

Suffix Automaton の性質

ノード	1	2	3	4	5	6	7
管理する 部分文字列	a	ab b	aba ba	abab bab	ababc babc abc bc c	ababcb babcb abcb bob cb	b

- さらに，“b” はノード 7 が管理する最も長い文字列（すなわち，T[7]）

Suffix Automaton の性質

ノード	1	2	3	4	5	6	7
管理する 部分文字列	a	ab	aba ba	abab bab	ababc babc abc bc c	ababcb babcb abcb bob cb	b

- 重要な性質:
  T[u] の接尾辞のうち, u が管理していない
  最長のもの(存在すれば)を X[u] で表す

Suffix Automaton の性質

ノード	1	2	3	4	5	6	7
管理する 部分文字列	a	ab	aba ba	abab bab	ababc babc abc bc c	ababcb babcb abcb bob cb	b

- 重要な性質:

  例: X[4] = “ab”, X[6] = “b”

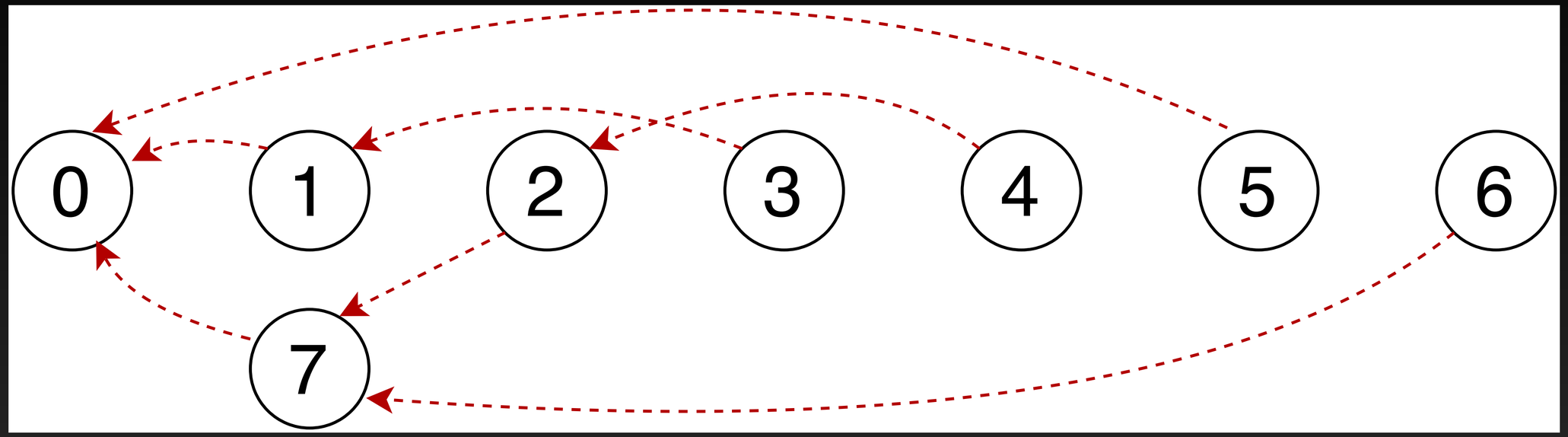
  X[u] = T[v] となる v がただ一つ存在する

Suffix Automaton の性質

ノード	1	2	3	4	5	6	7
管理する 部分文字列	a	ab b	aba ba	abab bab	ababc babc abc bc c	ababcb babcb abcb bob cb	b

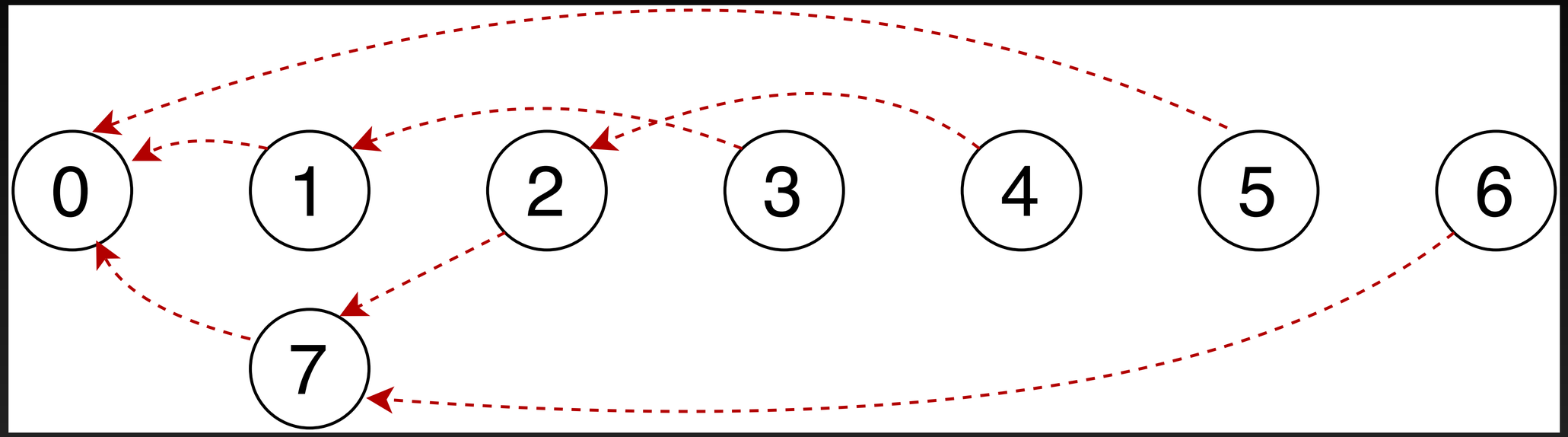
- このような u, v の関係を表現したい
  (例: 4 -> 2, 2 -> 7)

Suffix Automaton の性質



- 先ほど出てきた破線の部分
- Suffix Link という

Suffix Automaton の性質



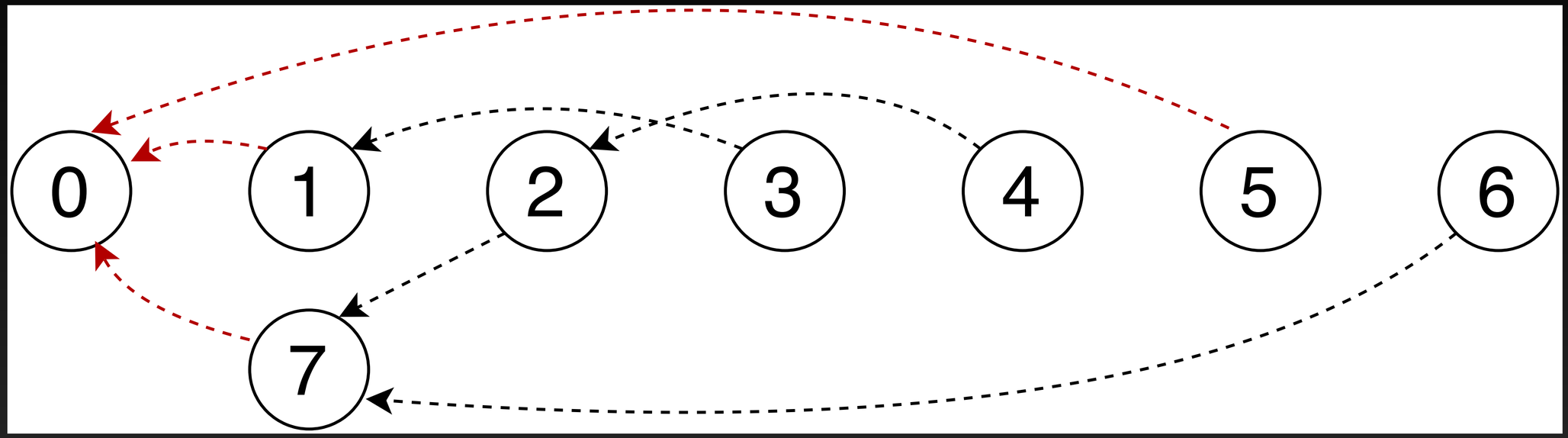
- 破線部分だけ取り出すと，ノード 0 を根とする根付き木になっている

Suffix Automaton の性質

ノード	1	2	3	4	5	6	7
管理する 部分文字列	a	ab	aba ba	abab bab	ababc babc abc bc c	ababcb babcb abcb bob cb	b

- u が T[u] の接尾辞全てを管理する場合:
  Suffix Link はノード 0 につながれる

Suffix Automaton の性質



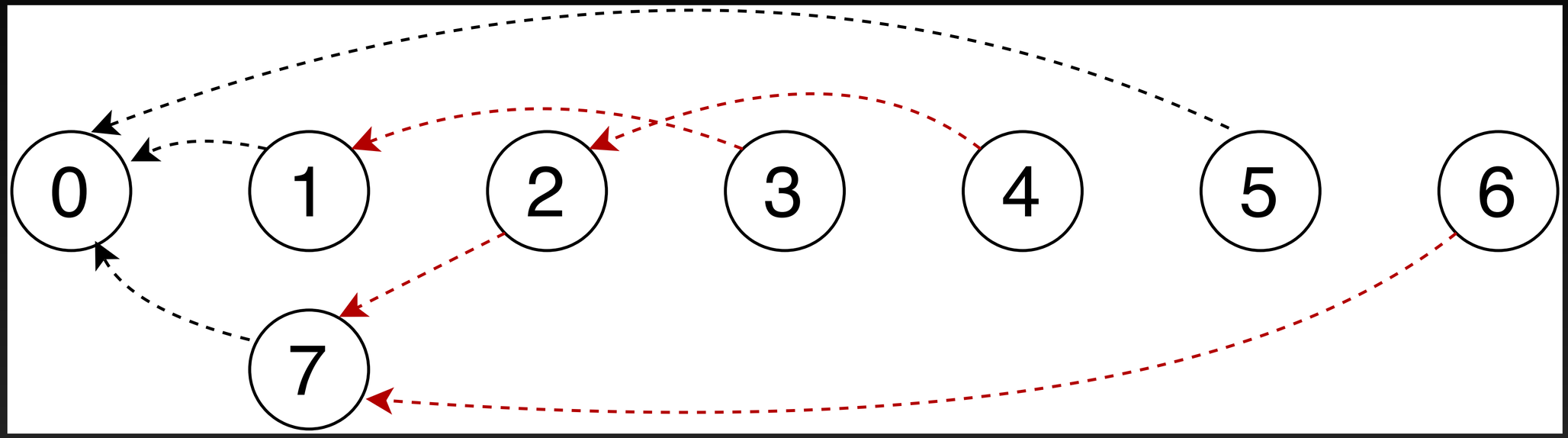
- u が T[u] の接尾辞全てを管理する場合:
  Suffix Link はノード 0 につながれる

Suffix Automaton の性質

ノード	1	2	3	4	5	6	7
管理する 部分文字列	a	ab	aba ba	abab bab	ababc babc abc bc c	ababcb babcb abcb bob cb	b

- それ以外: X[u] = T[v] となる v につながれる

Suffix Automaton の性質



- それ以外: X[u] = T[v] となる v につながれる

Suffix Automaton の性質

- いろいろと複雑

- 構築する方法を通して理解しましょう

- ▶ 1. Suffix Automaton の性質
- 2. 具体的な構築方法
- 3. 計算量などの解析
- 4. 実際に使ってみよう！

1. Suffix Automaton の性質

▶2. 具体的な構築方法

3. 計算量などの解析

4. 実際に使ってみよう！

具体的な構築方法

- 大まかな方針:
  - 現在の文字列が S[0]..S[n-1] だとして，新たに文字 c を付け足すことを考える
  - 始点から一番遠い頂点を u とすると，T[u] = S[0]..S[n - 1] であるはず．

1. Suffix Automaton の性質

▶2. 具体的な構築方法

3. 計算量などの解析

4. 実際に使ってみよう！

1. Suffix Automaton の性質

2. 具体的な構築方法

▶3. 計算量などの解析

4. 実際に使ってみよう！

計算量などの解析

- 文字列 S の長さを n とおく

- 頂点数:
  一文字追加するごとに高々 2 個しか増えない
  始点も入れて高々 2n+1 個

計算量などの解析

- 辺数:

1. Suffix Automaton の性質

2. 具体的な構築方法

▶3. 計算量などの解析

4. 実際に使ってみよう！

1. Suffix Automaton の性質

2. 具体的な構築方法

3. 計算量などの解析

▶4. 実際に使ってみよう！

実際に使ってみよう！

Q1. 文字列 S の部分文字列の種類数を求めよ.

<考察>

- Suffix Automaton におけるパスは,
  S の部分文字列を表している
- パスを数え上げればよい

実際に使ってみよう！

Q1. 文字列 S の部分文字列の種類数を求めよ.

<解法>

・トポロジカルソートする

・dp[始点] = 1 として配り，総和を求める

実際に使ってみよう！

Q2. 文字列 S の部分文字列のうち，
辞書順で K 番目のものを求めよ．

おわりに

・たのしかったです