Deduplication on Finite Automata and Nested Duplication Systems

Da-Jung Cho, Yo-Sub Han, and Hwee Kim

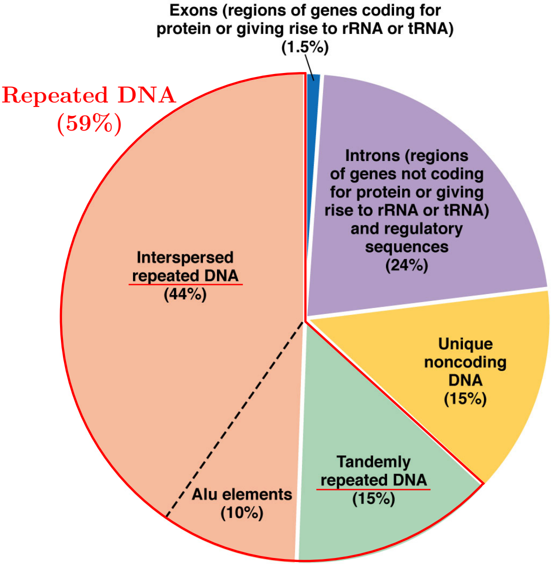
Department of Computer Science
Yonsei University

Unconventional Computation and Natural Computation, 2017

Outline

Repeated DNA in Human Genome

Human genome contains large amount of repeated DNA.



Researches on Duplications

In formal language theory, a DNA is a string over $\Sigma = \{A, G, C, T\}$.

- Duplication operation on strings and their properties
  - Searls, *"The computational linguistics of biological sequences"*, 1993.
  - Dassow et al., *"On the regularity of duplication closure"*, 1999.
  - Dassow et al., *"Operations and language generating devices suggested by the genome evolution"*, 2002.
  - Leupold et al., *"Formal languages arising from gene repeated duplication"*, 2003.
  - Leupold et al., *"Uniformly bounded duplication languages"*, 2005.
  - Ito et al., *"Duplication in DNA sequences"*, 2008.
  - Cho et al., *"Duplications and pseudo-duplications"*, 2016.
- Duplication Grammar
  - Martìn-Vide and Păun, *"Duplication grammars"*, 1999.
  - Mitrana and Rozenberg, *"Some properties of duplication grammars"*, 1999.

From Information Theory Viewpoint

Jain et al.[1] and Farnoud et al.[2] considered the string duplication systems and computed the capacity of the systems:

- *string duplication system* $S = (\Sigma, s, \tau)$ generates all strings by applying duplication function of τ to an initial string s a finite number times,
- *capacity* of string duplication system represents how many strings of length n that the system produces out of $|\Sigma^n|$, where n goes to infinity.

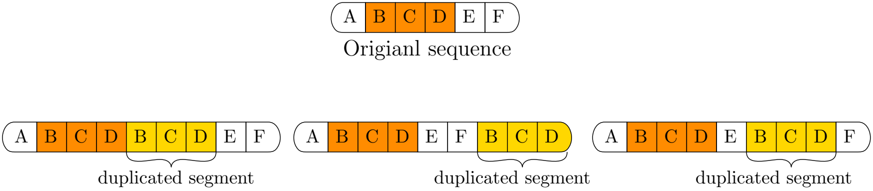
[1] S. Jain, F. Farnoud, and J. Bruck. *Capacity and expressiveness of genomic tandem duplication*, ISIT 2015

[2] F. Farnoud, M. Schwartz, and J. Bruck. *The capacity of string-duplication systems*, IEEE Transactions on Information Theory, 2016

From Information Theory Viewpoint

Jain et al. and Farnoud et al. considered the string duplication systems and computed the capacity of the systems:

- *string duplication system* $S = (\Sigma, s, \tau)$ generates all strings by applying duplication function of τ to an initial string s an infinity number times,
- *capacity* of string duplication system represents how many strings of length n that the system produces out of $|\Sigma^n|$, where n goes to infinity.



(a) Tandem duplication (b) End duplication (c) Interspersed duplication

From Information Theory Viewpoint

Jain et al. and Farnoud et al. considered the string duplication systems and computed the capacity of the systems:

- *string duplication system* generates all strings by applying duplication function to an initial string an infinity number times,
- *capacity* of string duplication system represents how many strings of length n that the system produces out of $|\Sigma^n|$, where n goes to infinity:

$$cap(S) = \lim_{n \to \infty} \sup \frac{\log_{|\Sigma|} |S \cap \Sigma^n|}{n}$$

From Information Theory Viewpoint

The *Capacity* of string duplication system represents how many strings of length n the system produces out of $|\Sigma^n|$, where n goes to infinity.

Example

From a string 01 over $\Sigma = \{0, 1\}$, a system duplicates any substring of length up to 2 from 01 repetitively.

- $01 \rightarrow 011 \rightarrow 01111 \rightarrow 0101111...$

Then, the number of strings of length n is 2^{n-2}.

Encoding strings of length n in this system need $n-2$ bits.

The average number of bits per symbol = 1

From Information Theory Viewpoint

The *Capacity* of string duplication system represents how many strings of length n the system produces out of $|\Sigma^n|$, where n goes to infinity.

Example

From a string 01 over $\Sigma = \{0, 1\}$, a system duplicates any substring of length up to 2 from 01 repetitively.

- $01 \rightarrow 011 \rightarrow 01111 \rightarrow 0101111...$

Then, the number of strings of length n is 2^{n-2}.

Encoding strings of length n in this system need $n-2$ bits.

The capacity of this system = 1

From Information Theory Viewpoint

For computing the capacity, $\lim_{n\to\infty} \sup \frac{\log_{|\Sigma|} |S \cap \Sigma^n|}{n}$, of the duplication system $S = (\Sigma, s, \tau)$, we need to consider all strings

- of length n, where n goes to infinity,
- iteratively obtained by a duplication function $\mathbb{D} \in \tau$ from an initial string s,

$$\mathbb{D}^i(\mathbb{D}^{i-1}(\cdots(\mathbb{D}^2(\mathbb{D}^1(s))))), \text{ for } i \geq 1.$$

Jain et al. (2015)

Given a finite directed graph that represents the string duplication system S, we can compute the capacity of S using *Perron-Frobenus Theory* [Lind and Marcus, 1985].

It is hard to construct a finite automata!

Our Goal from Formal Language Theory

1. Define a variant of duplication functions named *nested tandem duplication*
2. Construct an NFA for a set of strings obtained by a nested duplication system
   - NFA construction (possible for any restriction on length of duplicated substring)
   - $\mathbb{D}$-cycle deduplication on NFA
3. Compute the capacity of the nested duplication system

Definition of Nested Duplication

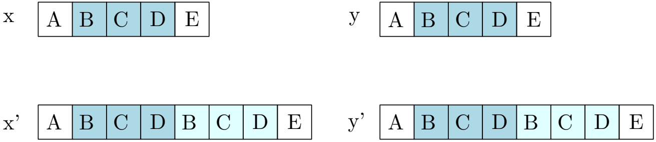
Definition (Tandem Duplication)

A tandem duplication $\mathbb{D}^{tan}_{\leq k}$ of x is defined

$$\mathbb{D}^{tan}_{\leq k}(x) = \begin{cases} uvvw & \text{for } x = uvw, |u| = i, |v| \leq k \\ x & \text{otherwise.} \end{cases}$$

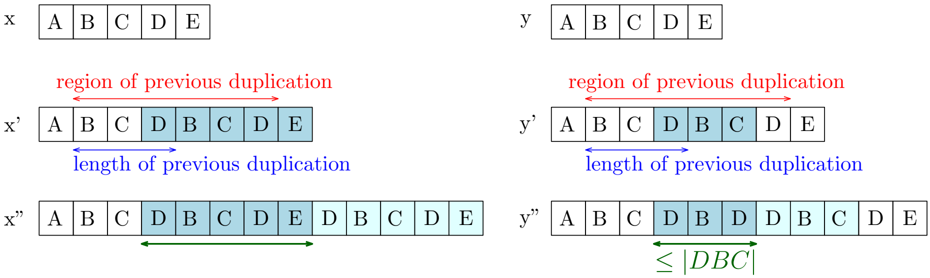
For a string x, $\mathbb{D}^{tan}_{\leq k}(x)$ allows a substring v of length up to k starting at position $i+1$ to be duplicated next to its original position.

Tandem Duplication and Nested Duplication



(a) A tandem duplication (b) A nested duplication

Tandem Duplication and Nested Duplication



(a) A tandem duplication

(b) A nested duplication

Definition of Nested Duplication

Definition (Nested Duplication)

A nested duplication $\mathbb{D}^{nes}_{\leq k}$ of x with a array d is defined $\mathbb{D}^{nes}_{\leq k}(x, d)$=

$$\begin{cases} (uvvw, d[1 : |u|] \cdot |v|^{2|v|} \cdot d[|u|+|v|+1 : |x|]) & \text{if } d[j] \geq |v| \\ & \text{for } |u|+1 \leq j \leq |u|+|v|, \\ (x, d) & \text{otherwise.} \end{cases}$$

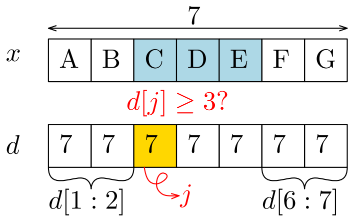
- For a string $w = w_1 w_2 \cdots w_n$ and $i \geq 1$, $w[1 : i]$ denotes a substring $w_1 \cdots w_i$.
- For an integer $j \geq 1$, j^k denotes k consecutive j.

Definition of Nested Duplication

Definition (Nested Duplication)

A nested duplication $\mathbb{D}^{nes}_{\leq k}$ of x with a array d is defined $\mathbb{D}^{nes}_{\leq k}(x, d)=$

$$\begin{cases} (uvvw, d[1 : |u|] \cdot |v|^{2|v|} \cdot d[|u|+|v|+1 : |x|]) & \text{if } d[j] \geq |v| \\ & \text{for } |u|+1 \leq j \leq |u|+|v|, \\ (x, d) & \text{otherwise.} \end{cases}$$



Definition of Nested Duplication

Definition (Nested Duplication)

A nested duplication $\mathbb{D}^{nes}_{\leq k}$ of x with a array d is defined $\mathbb{D}^{nes}_{\leq k}(x, d)$=

$$\begin{cases} (uvvw, d[1:|u|] \cdot |v|^{2|v|} \cdot d[|u|+|v|+1:|x|]) & \text{if } d[j] \geq |v| \text{ for } |u|+1 \leq j \leq |u|+|v|, \\ (x, d) & \text{otherwise.} \end{cases}$$

$$(y, d') \in \mathbb{D}^{nes}_{2,3}(x, d)$$

y	A	B	C	D	E	C	D	E	F	G
d'	7	7	3	3	3	3	3	3	7	7

$d[1:2]$ $\quad$ 3^{3*2} $\quad$ $d[2+3+1:7]$

Definition of Nested Duplication

Definition (Nested Duplication)

A nested duplication $\mathbb{D}^{nes}_{\leq k}$ of x with a array d is defined $\mathbb{D}^{nes}_{\leq k}(x,d)$=

$$\begin{cases} (uvvw, d[1:|u|]\cdot|v|^{2|v|}\cdot d[|u|+|v|+1:|x|]) & \text{if } d[j] \geq |v| \\ & \text{for } |u|+1 \leq j \leq |u|+|v|, \\ (x,d) & \text{otherwise.} \end{cases}$$

y	A	B	C	D	E	C	D	E	F	G
d'	7	7	3	3	3	3	3	3	7	7

$d'[j] \geq 4?$

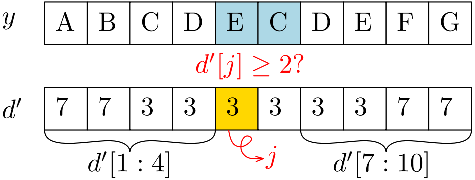
j

Definition of Nested Duplication

Definition (Nested Duplication)

A nested duplication $\mathbb{D}^{nes}_{\leq k}$ of x with a array d is defined $\mathbb{D}^{nes}_{\leq k}(x, d)$=

$$\begin{cases} (uvvw, d[1:|u|] \cdot |v|^{2|v|} \cdot d[|u|+|v|+1:|x|]) & \text{if } d[j] \geq |v| \\ & \text{for } |u|+1 \leq j \leq |u|+|v|, \\ (x, d) & \text{otherwise.} \end{cases}$$



Definition of Nested Duplication

Definition (Nested Duplication)

A nested duplication $\mathbb{D}^{nes}_{\leq k}$ of x with a array d is defined $\mathbb{D}^{nes}_{\leq k}(x, d)$=

$$\begin{cases} (uvvw, d[1 : |u|] \cdot |v|^{2|v|} \cdot d[|u|+|v|+1 : |x|]) & \text{if } d[j] \geq |v| \\ & \text{for } |u|+1 \leq j \leq |u|+|v|, \\ (x, d) & \text{otherwise.} \end{cases}$$

$$(z, d'') \in \mathbb{D}^{nes}_{\leq 3}(z, d'')$$

z	A	B	C	D	E	C	E	C	D	E	F	G
d''	7	7	3	3	2	2	2	2	3	3	7	7

$d'[1:4]$ 2^{2*2} $d'[7:10]$

Nested Duplication System

Definition (Nested Duplication System)

A *nested duplication system* consists of three tuples $S = (\Sigma, s, \mathbb{D}^{nes}_{\leq k})$,

- $s \in \Sigma^*$ is seed, a finite length string,
- $\mathbb{D}^{nes}_{\leq k}$ is the nested duplication function.

We call the set of all strings generated by the system $S = (\Sigma, s, \mathbb{D}^{nes}_{\leq k})$

the language generated by S, $L(S)$.

Nested Duplication System

Definition (Nested Duplication System)

A *nested duplication system* consists of three tuples $S = (\Sigma, s, \mathbb{D}^{nes}_{\leq k})$,

- $s \in \Sigma^*$ is seed, a finite length string,
- $\mathbb{D}^{nes}_{\leq k}$ is the nested duplication function.

We call the set of all strings generated by the system $S = (\Sigma, s, \mathbb{D}^{nes}_{\leq k})$

the language generated by S, $L(S)$.

We extend the nested duplication system to a language

$$S = (\Sigma, L, \mathbb{D}^{nes}_{\leq k}).$$

NFA Construction for $L(S)$

We construct an NFA M_S recognizing the language $L(S)$ generated by the nested duplication system $S = (\Sigma, s, \mathbb{D}^{nes}_{\leq k})$.

NFA Construction for $L(S)$

An NFA M_S is a tuple $(Q, \Sigma, \delta, 0_0, \{n_0\})$, where

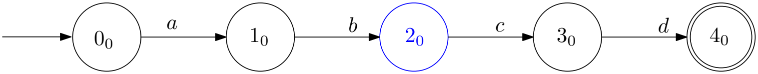
$$Q = \{q \mid q = l[1]_{d[1]} l[2]_{d[2]} \cdots l[t]_{d[t]}, 1 \leq t \leq n, d[1] = 0\}$$

- t represents the depth of a duplication, $1 \leq t \leq n$
- for each depth t, $d[t]$ denotes the length of duplications,
- $l[t]$ represents $\#$ of characters duplicated so far by the last duplication in depth t.

NFA Construction for $L(S)$

$S = (\Sigma, abcd, \mathbb{D}^{nes}_{\leq 4})$

$q = l[1]_{d[1]} l[2]_{d[2]} \cdots l[t]_{d[t]}$



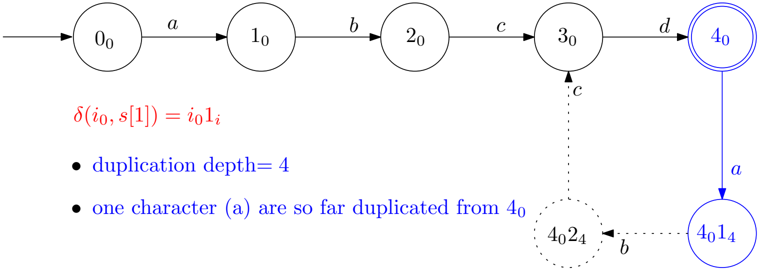
$\delta((i-1)_0, s[i]) = i_0$

- duplication depth$= 0$
- two characters are so far read

NFA Construction for $L(S)$

$S = (\Sigma, abcd, \mathbb{D}^{nes}_{\leq 4})$

$q = l[1]_{d[1]} l[2]_{d[2]} \cdots l[t]_{d[t]}$



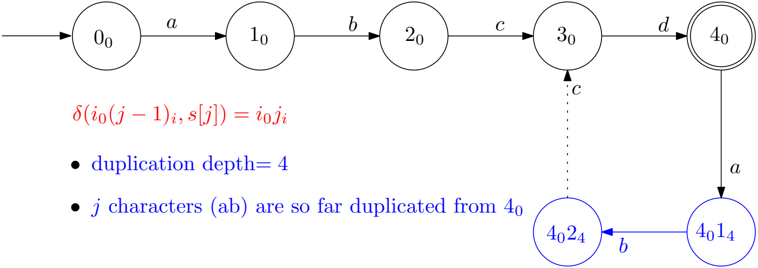
$\delta(i_0, s[1]) = i_0 1_i$

- duplication depth= 4
- one character (a) are so far duplicated from 4_0

NFA Construction for $L(S)$

$S = (\Sigma, abcd, \mathbb{D}^{nes}_{\leq 4})$

$q = l[1]_{d[1]} l[2]_{d[2]} \cdots l[t]_{d[t]}$



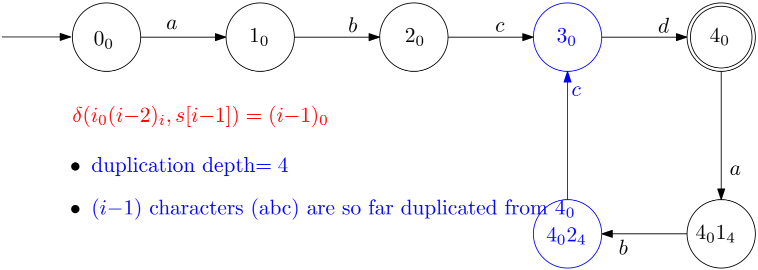
$\delta(i_0(j-1)_i, s[j]) = i_0 j_i$

- duplication depth $= 4$
- j characters (ab) are so far duplicated from 4_0

NFA Construction for $L(S)$

$S = (\Sigma, abcd, \mathbb{D}^{nes}_{\leq 4})$

$q = l[1]_{d[1]} l[2]_{d[2]} \cdots l[t]_{d[t]}$



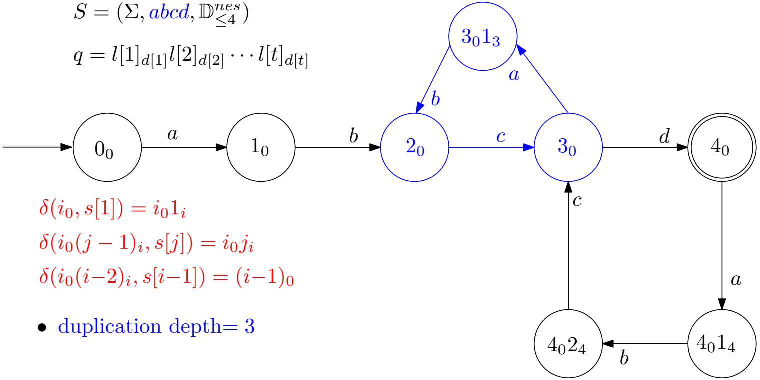
$\delta(i_0(i-2)_i, s[i-1]) = (i-1)_0$

- duplication depth= 4
- $(i-1)$ characters (abc) are so far duplicated from 4_0

NFA Construction for $L(S)$

$S = (\Sigma, abcd, \mathbb{D}^{nes}_{\leq 4})$

$q = l[1]_{d[1]} l[2]_{d[2]} \cdots l[t]_{d[t]}$



$\delta(i_0, s[1]) = i_0 1_i$

$\delta(i_0 (j-1)_i, s[j]) = i_0 j_i$

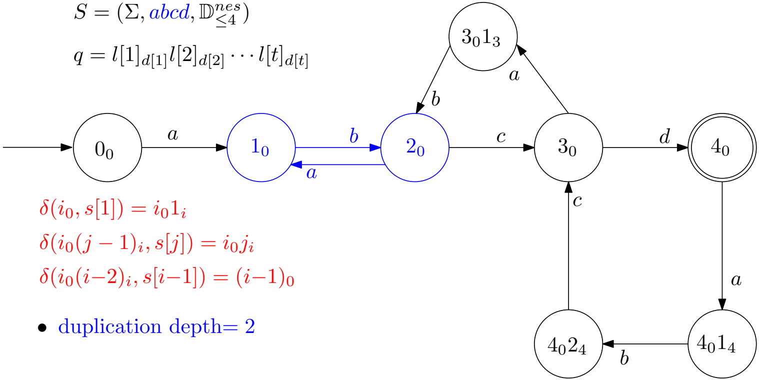
$\delta(i_0 (i-2)_i, s[i-1]) = (i-1)_0$

- duplication depth= 3

NFA Construction for $L(S)$

$S = (\Sigma, abcd, \mathbb{D}^{nes}_{\leq 4})$

$q = l[1]_{d[1]} l[2]_{d[2]} \cdots l[t]_{d[t]}$



$\delta(i_0, s[1]) = i_0 1_i$

$\delta(i_0 (j-1)_i, s[j]) = i_0 j_i$

$\delta(i_0 (i{-}2)_i, s[i{-}1]) = (i{-}1)_0$

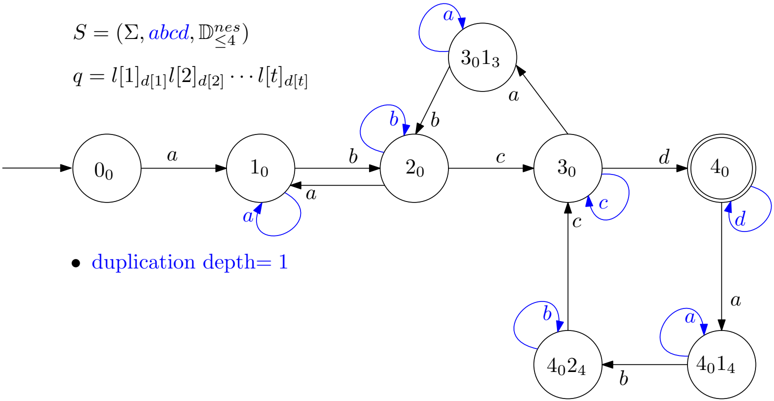
- duplication depth= 2

NFA Construction for $L(S)$

$S = (\Sigma, abcd, \mathbb{D}^{nes}_{\leq 4})$

$q = l[1]_{d[1]} l[2]_{d[2]} \cdots l[t]_{d[t]}$

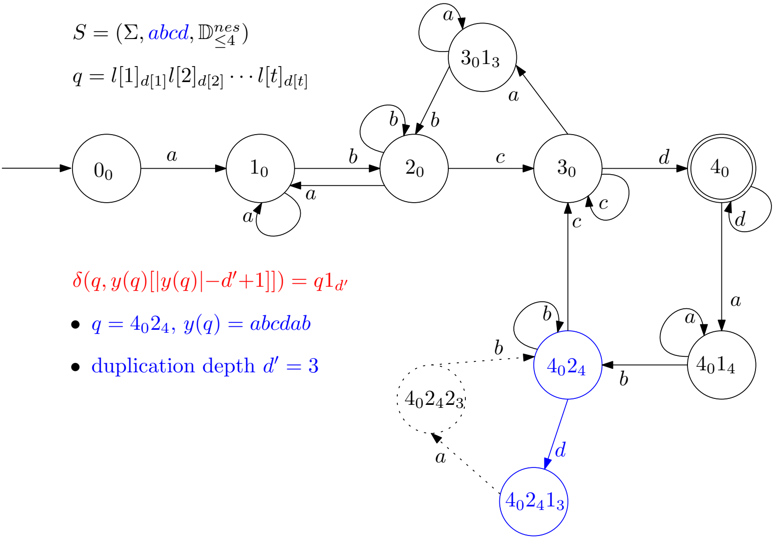
- duplication depth$= 1$



NFA Construction for $L(S)$

$S = (\Sigma, abcd, \mathbb{D}^{nes}_{\leq 4})$

$q = l[1]_{d[1]} l[2]_{d[2]} \cdots l[t]_{d[t]}$



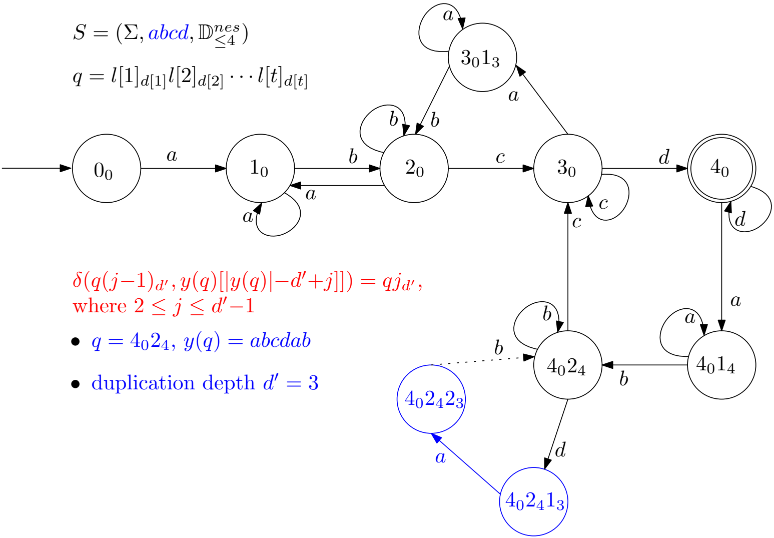
$\delta(q, y(q)[|y(q)| - d' + 1]) = q1_{d'}$

- $q = 4_0 2_4,\ y(q) = abcdab$
- duplication depth $d' = 3$

NFA Construction for $L(S)$

$S = (\Sigma, abcd, \mathbb{D}^{nes}_{\leq 4})$

$q = l[1]_{d[1]} l[2]_{d[2]} \cdots l[t]_{d[t]}$



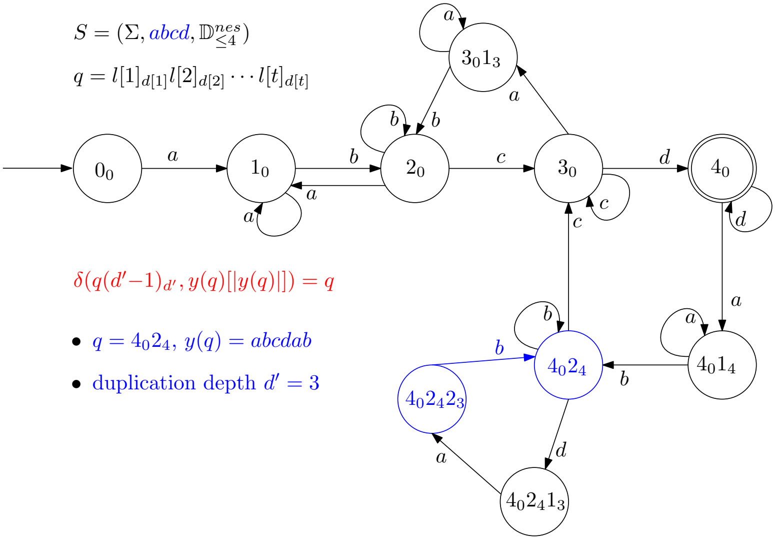
$\delta(q(j-1)_{d'}, y(q)[|y(q)|-d'+j]) = qj_{d'}$,
where $2 \leq j \leq d'-1$

- $q = 4_0 2_4$, $y(q) = abcdab$
- duplication depth $d' = 3$

NFA Construction for $L(S)$

$S = (\Sigma, abcd, \mathbb{D}^{nes}_{\leq 4})$

$q = l[1]_{d[1]} l[2]_{d[2]} \cdots l[t]_{d[t]}$



$\delta(q(d'-1)_{d'}, y(q)[|y(q)|]) = q$

- $q = 4_0 2_4$, $y(q) = abcdab$
- duplication depth $d' = 3$

NFA Construction for $L(S)$

$S = (\Sigma, abcd, \mathbb{D}^{nes}_{\leq 4})$

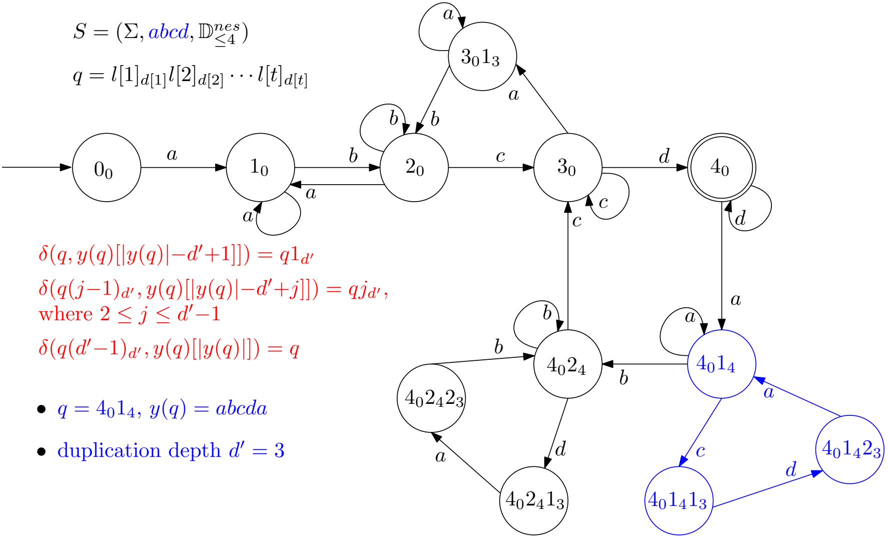
$q = l[1]_{d[1]} l[2]_{d[2]} \cdots l[t]_{d[t]}$

$\delta(q, y(q)[|y(q)| - d' + 1]]) = q1_{d'}$

$\delta(q(j-1)_{d'}, y(q)[|y(q)| - d' + j]]) = qj_{d'}$, where $2 \leq j \leq d' - 1$

$\delta(q(d'-1)_{d'}, y(q)[|y(q)|]) = q$

- $q = 4_0 1_4$, $y(q) = abcda$
- duplication depth $d' = 3$



NFA Construction for $L(S)$

$S = (\Sigma, abcd, \mathbb{D}^{nes}_{\leq 4})$

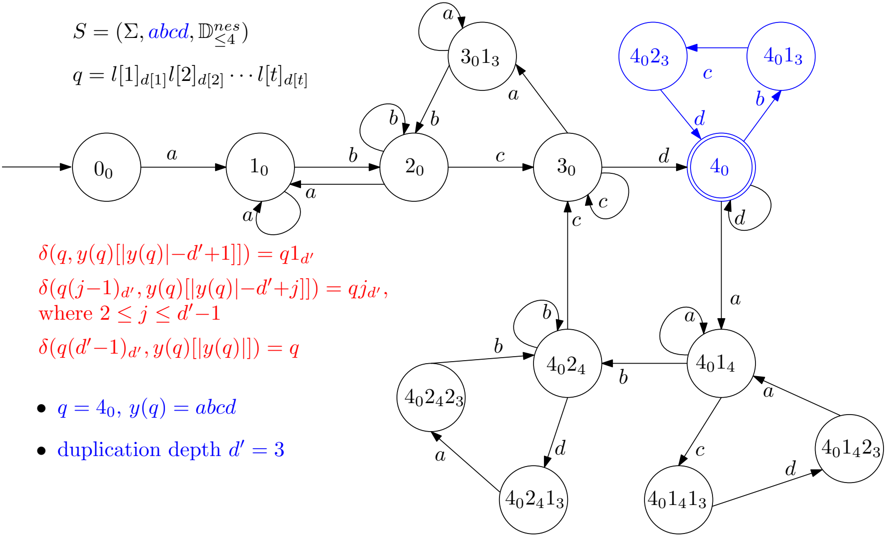
$q = l[1]_{d[1]} l[2]_{d[2]} \cdots l[t]_{d[t]}$

$\delta(q, y(q)[|y(q)| - d' + 1]]) = q1_{d'}$

$\delta(q(j-1)_{d'}, y(q)[|y(q)| - d' + j]]) = qj_{d'}$,
where $2 \leq j \leq d' - 1$

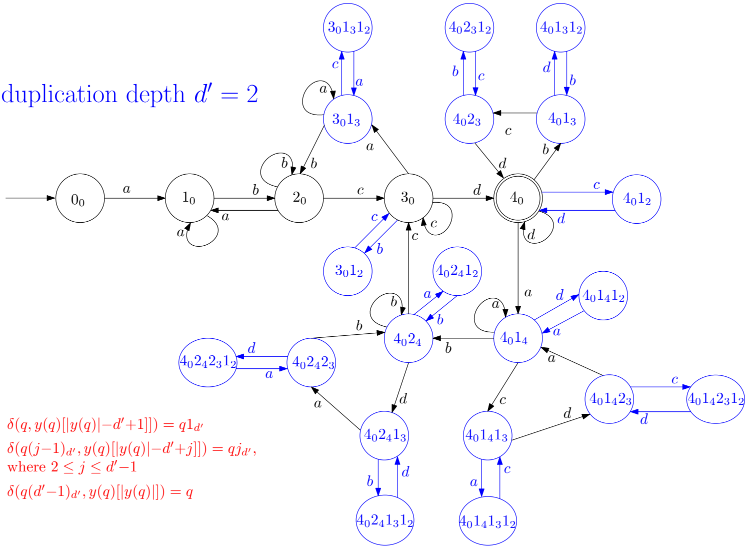
$\delta(q(d'-1)_{d'}, y(q)[|y(q)|]) = q$

- $q = 4_0$, $y(q) = abcd$
- duplication depth $d' = 3$



NFA Construction for $L(S)$

duplication depth $d' = 2$



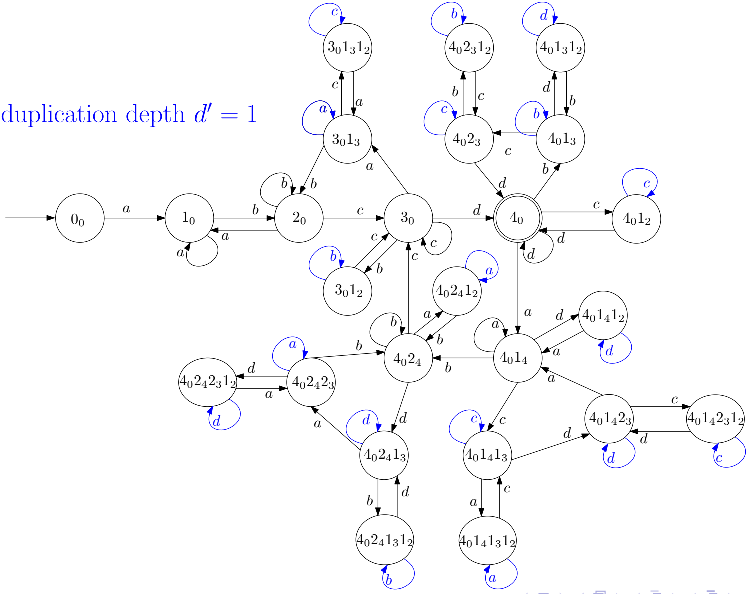
$\delta(q, y(q)[|y(q)|-d'+1]]) = q1_{d'}$

$\delta(q(j-1)_{d'}, y(q)[|y(q)|-d'+j]]) = qj_{d'}$,
where $2 \leq j \leq d'-1$

$\delta(q(d'-1)_{d'}, y(q)[|y(q)|]]) = q$

NFA Construction for $L(S)$

duplication depth $d' = 1$



NFA Construction for $L(S)$

Given a set $\mathcal{L}$ of possible duplication lengths,

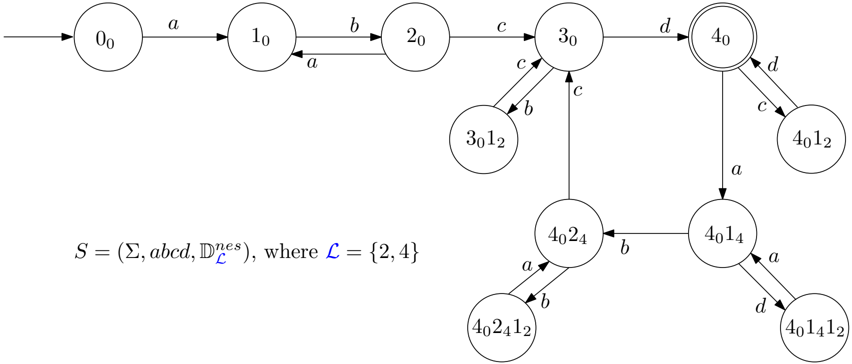


Figure: An example of NFA recognizing $L(S)$, where $S = (\Sigma, abcd, \mathbb{D}_{\mathcal{L}}^{nes})$.

NFA Construction for $L(S)$

Theorem

The NFA $M_S = (Q, \Sigma, \delta, 0_0, \{n_0\})$ recognizes the language generated by the nested duplication system $S = (\Sigma, s, \mathbb{D}^{nes}_{\leq k})$, $L(M_S) = L(S)$.

- **If $x \in L(S)$, then $x \in L(M_S)$.**
- **If $x \in L(M_S)$, then $x \in L(S)$.**

NFA Construction for $L(S)$

Theorem

The NFA $M_S = (Q, \Sigma, \delta, 0_0, \{n_0\})$ recognizes the language generated by the nested duplication system $S = (\Sigma, s, \mathbb{D}^{nes}_{\leq k})$, $L(M_S) = L(S)$.

If $x \in L(M_S)$, then $x \in L(S)$.

- For a string $x \in L(M_S)$, let p be the path that yields x.
- We recursively generate a series of paths p_i and strings x_i:
  1. p_1: generated by removing all self loops in p.
  2. ...
  3. p_i: generated by removing all cycles of size i in p_{i-1}.
- Then, $x_n = s$.

NFA Construction for $L(S)$

Theorem

The NFA $M_S = (Q, \Sigma, \delta, 0_0, \{n_0\})$ recognizes the language generated by the nested duplication system $S = (\Sigma, s, \mathbb{D}^{nes}_{\leq k})$, $L(M_S) = L(S)$.

If $x \in L(M_S)$, then $x \in L(S)$.

- For a string $x \in L(M_S)$, let p be the path that yields x.
- We recursively generate a series of paths p_i and strings x_i:
  1. p_1: generated by removing all self loops in p.
  2. ...
  3. p_i: generated by removing all cycles of size i in p_{i-1}.
- Then, $x_n = s$.

Deduplication on finite automata!

Deduplication on Finite Automata

For a string $w' = xyyz$, where $|y| \leq k$, deduplication of w' transforms yy into y,

$$xyyz \rightarrow xyz.$$

$\mathbb{D}$-cycle deduplication

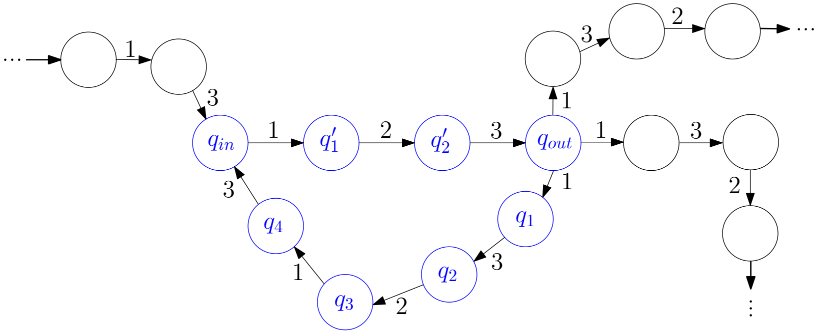
$\mathbb{D}$*-cycle deduplication* transforms a given NFA M to a smaller NFA M' while generating the same language in the duplication system by removing cycles in the NFA that satisfies special conditions,

$$L(S) = L(S'),$$

where $S = (\Sigma, L(M), \mathbb{D}^{nes}_{\leq k})$ and $S' = (\Sigma, L(M'), \mathbb{D}^{nes}_{\leq k})$.

$\mathbb{D}$-cycle deduplication

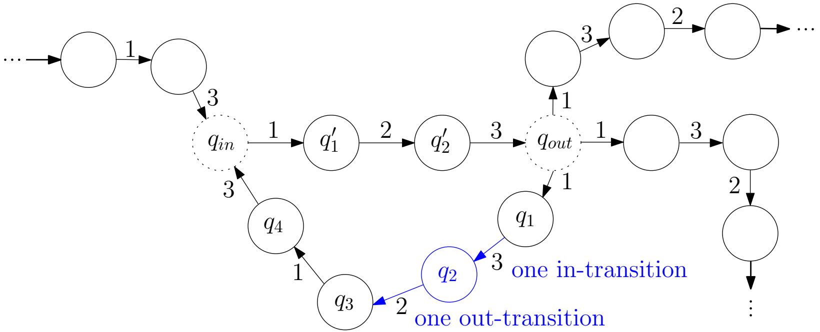
For a cycle C in an NFA, we call the cycle $\mathbb{D}$-*cycle* if C satisfies special conditions:



The cycle $C = (q_{in}, q'_1, q'_2, q_{out}, q_1, q_2, q_3, q_4, q_{in})$

$\mathbb{D}$-cycle deduplication

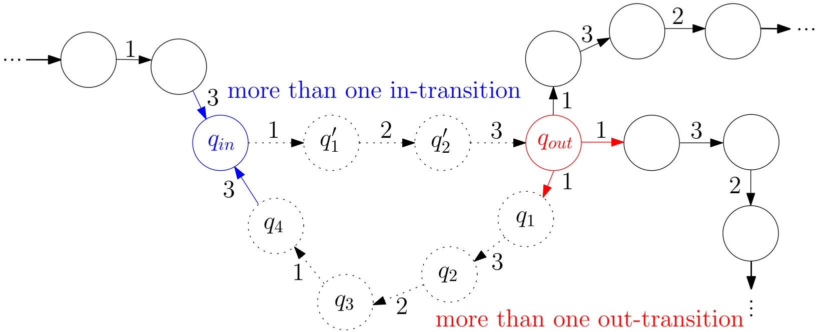
For a cycle C in an NFA, we call the cycle $\mathbb{D}$-*cycle* if C satisfies special conditions:



The cycle $C = (q_{in}, q'_1, q'_2, q_{out}, q_1, q_2, q_3, q_4, q_{in})$

$\mathbb{D}$-cycle deduplication

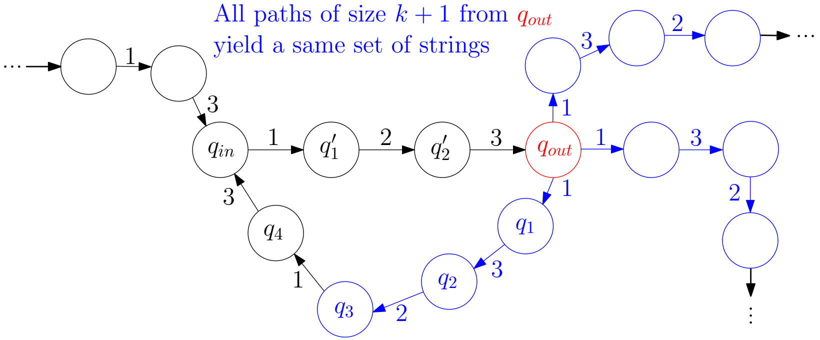
For a cycle C in an NFA, we call the cycle $\mathbb{D}$-*cycle* if C satisfies special conditions:



The cycle $C = (q_{in}, q'_1, q'_2, q_{out}, q_1, q_2, q_3, q_4, q_{in})$

$\mathbb{D}$-cycle deduplication

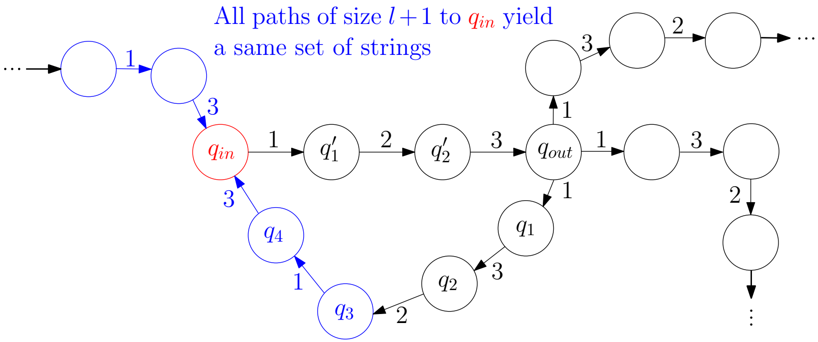
For a cycle C in an NFA, we call the cycle $\mathbb{D}$-*cycle* if C satisfies special conditions:



The cycle $C = (q_{in}, q'_1, q'_2, q_{out}, q_1, q_2, q_3, q_4, q_{in})$

$\mathbb{D}$-cycle deduplication

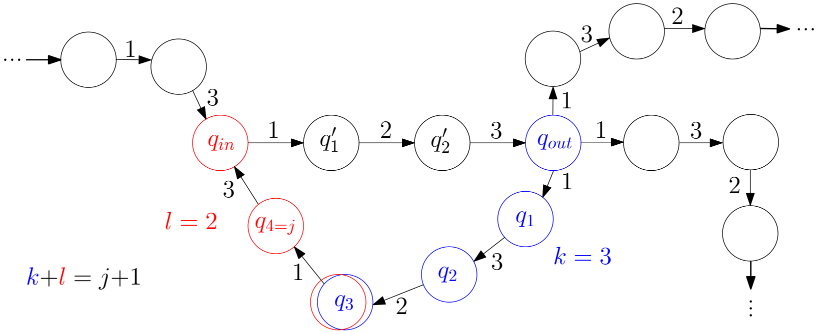
For a cycle C in an NFA, we call the cycle $\mathbb{D}$-*cycle* if C satisfies special conditions:



The cycle $C = (q_{in}, q'_1, q'_2, q_{out}, q_1, q_2, q_3, q_4, q_{in})$

$\mathbb{D}$-cycle deduplication

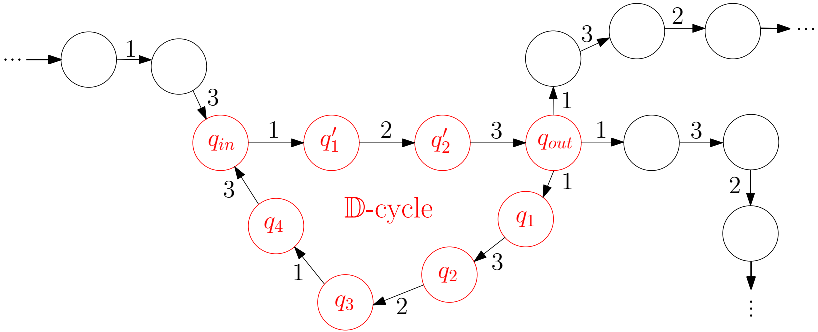
For a cycle C in an NFA, we call the cycle $\mathbb{D}$-*cycle* if C satisfies special conditions:



The cycle $C = (q_{in}, q'_1, q'_2, q_{out}, q_1, q_2, q_3, q_4, q_{in})$

$\mathbb{D}$-cycle deduplication

For a cycle C in an NFA, we call the cycle $\mathbb{D}$-*cycle* if C satisfies special conditions:



The $\mathbb{D}$-cycle $C = (q_{in}, q'_1, q'_2, q_{out}, q_1, q_2, q_3, q_4, q_{in})$

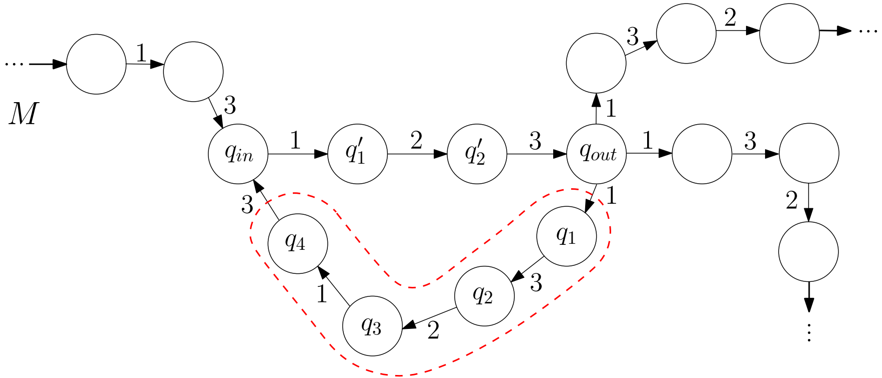
$\mathbb{D}$-cycle deduplication

Definition

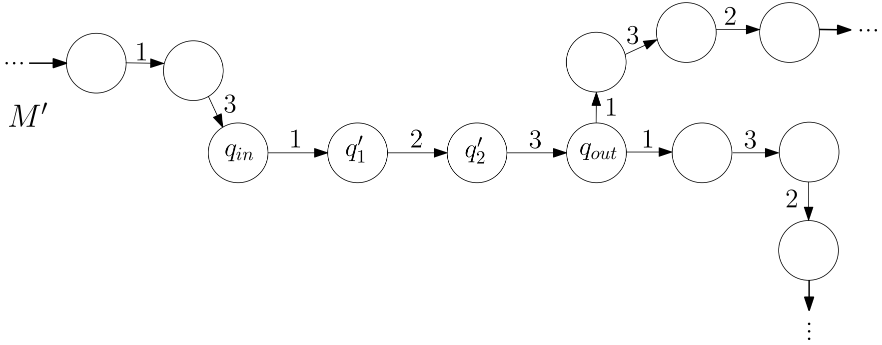
Given an NFA $M = (Q, \Sigma, \delta, s, F)$ with a $\mathbb{D}$-cycle $(q_{in}, q'_1, q'_2, \ldots, q'_i, q_{out}, q_1, q_2, \ldots, q_j, q_{in})$, we define a $\mathbb{D}$-cycle deduplication by $M \xrightarrow{\mathbb{D}^{-1}_{\leq h}} M'$, where $i+j = h$, to be

$$M' = (Q \backslash \{q_1, q_2, \ldots, q_j\}, \Sigma, \{\delta(p, \alpha) = q \mid p, q \notin \{q_1, q_2, \ldots, q_j\}\}, s, F).$$

$\mathbb{D}$-cycle deduplication



$\mathbb{D}$-cycle deduplication



Lemma

Given an NFA M and its deduplication M′ such that $M \xrightarrow{\mathbb{D}^{-1}_{\leq h}} M'$*,*
let $S = (\Sigma, L(M), \mathbb{D}^{nes}_{\leq k})$ *and* $S' = (\Sigma, L(M'), \mathbb{D}^{nes}_{\leq k})$*. Then,*

$$L(S) = L(S').$$

NFA Construction for $L(S)$

Theorem

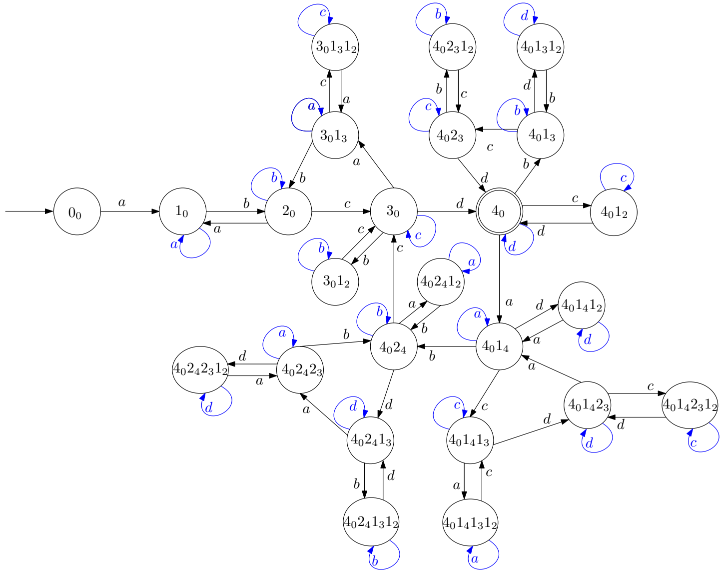
The NFA $M_S = (Q, \Sigma, \delta, 0_0, \{n_0\})$ recognizes the language generated by the nested duplication system $S = (\Sigma, s, \mathbb{D}^{nes}_{\leq k})$, $L(M_S) = L(S)$.

If $x \in L(M_S)$, then $x \in L(S)$.

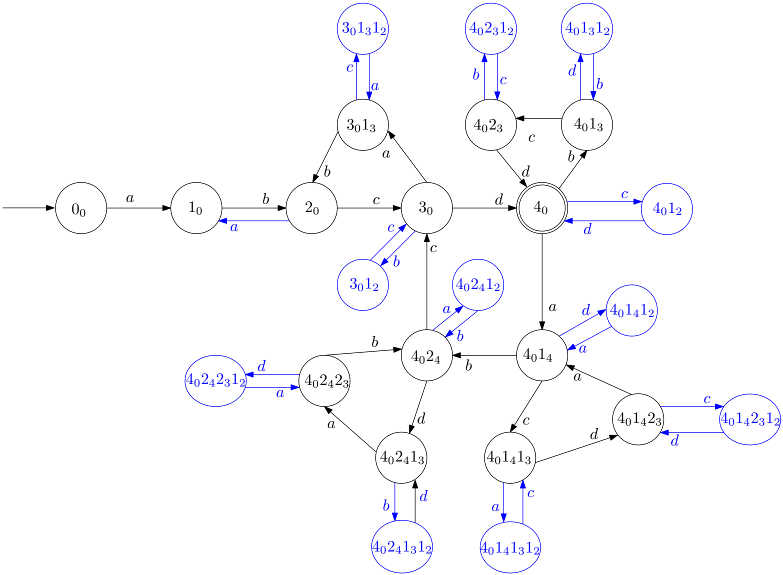
- For a string $x \in L(M_S)$, let p be the path that yields x.
- We recursively generate a series of paths p_i and strings x_i:
  1. p_1: generated by removing all self loops in p.
  2. ...
  3. p_i: generated by removing all cycles of size i in p_{i-1}.
- Then, $x_n = s$.

Deduplication on finite automata!

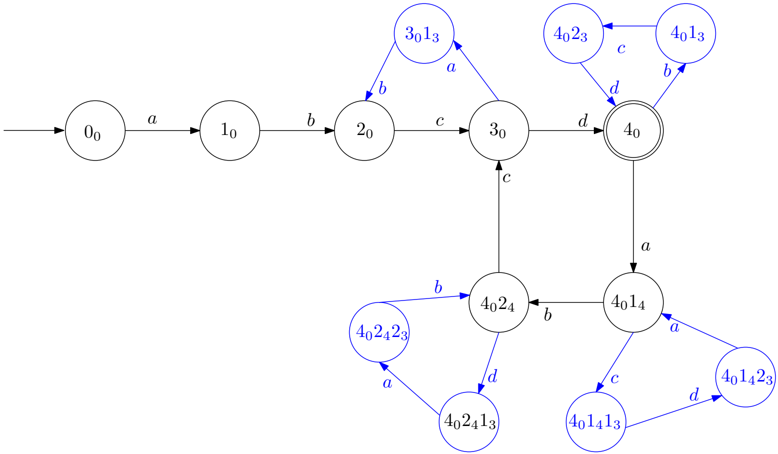
NFA Construction for $L(S)$



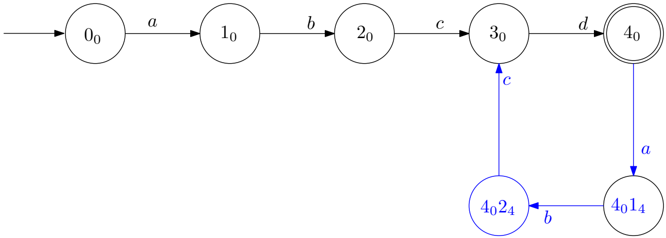
NFA Construction for $L(S)$



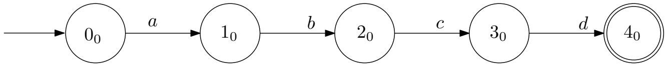
NFA Construction for $L(S)$



NFA Construction for $L(S)$



NFA Construction for $L(S)$



Computing the Capacity of S

Definition

The *capacity* of a duplication system S represents how many strings the system produces compared to Σ^n, where n goes to infinity,

$$cap(S) = \lim_{n\to\infty} \sup \frac{\log_{|\Sigma|} |S \cap \Sigma^n|}{n}.$$

From Jain et al. (2015), it is known that we can compute the capacity of S represented by DFA using Perron-Frobenius Theory.

Computing the Capacity of S

1. Construct an NFA M_S for $S = (\Sigma, s, \mathbb{D}^{nes}_{\leq k})$.
2. Convert M_S to a DFA M'.
3. Find the maximal connected component in M' and compute its adjacency matrix $\mathbb{M}$.
4. Return the maximum eigenvalue of $\mathbb{M}$ using Perron-Frobenius Theory.

Summary

- Defined
  - the nested duplication operation $\mathbb{D}^{nes}_{\leq k}(w)$,
  - the nested duplication system $S(\Sigma, s, \mathbb{D}^{nes}_{\leq k})$.
- Presented an NFA construction for L of $S(\Sigma, s, \mathbb{D}^{nes}_{\leq k})$
- Introduced the $\mathbb{D}$-cycle deduplication on NFA

☞ dajungcho.dothome.co.kr
☞ toc.yonsei.ac.kr

Thank you!