

String Matching

Simon Moura

26 february 2019

University College London

s.moura@ucl.ac.uk

Slides: https://github.com/simmou/ucl_algo

Problem definition

The string matching problem

We consider a text T and a pattern P , where

- A **text** $T = t_1, t_2, \dots, t_n$ is an array of n characters
- A **pattern** $P = p_1, p_2, \dots, p_m$ is an array of m consecutive characters
- $n > m$

The goal is to find **ALL** occurrences of pattern P in text T .

The string matching problem

We consider a text T and a pattern P , where

- A **text** $T = t_1, t_2, \dots, t_n$ is an array of n characters
- A **pattern** $P = p_1, p_2, \dots, p_m$ is an array of m consecutive characters
- $n > m$

The goal is to find ALL occurrences of pattern P in text T .

We say that the pattern P occurs with **shift** s in text T if the **substring** of T that starts at $t_{s+1} = P$

- i.e. $t_{s+1} = p_1, t_{s+2} = p_2, \dots, t_{s+m} = p_m$

The string matching problem

We consider a text T and a pattern P , where

- A **text** $T = t_1, t_2, \dots, t_n$ is an array of n characters
- A **pattern** $P = p_1, p_2, \dots, p_m$ is an array of m consecutive characters
- $n > m$

The goal is to find ALL occurrences of pattern P in text T .

We say that the pattern P occurs with **shift** s in text T if the **substring** of T that starts at $t_{s+1} = P$

- i.e. $t_{s+1} = p_1, t_{s+2} = p_2, \dots, t_{s+m} = p_m$

Concrete examples:

- The pattern $P = \text{ell}$ appears in the text $T = \text{Hello, world}$ with a shift 1.

The string matching problem

We consider a text T and a pattern P , where

- A **text** $T = t_1, t_2, \dots, t_n$ is an array of n characters
- A **pattern** $P = p_1, p_2, \dots, p_m$ is an array of m consecutive characters
- $n > m$

The goal is to find ALL occurrences of pattern P in text T .

We say that the pattern P occurs with **shift** s in text T if the **substring** of T that starts at $t_{s+1} = P$

- i.e. $t_{s+1} = p_1, t_{s+2} = p_2, \dots, t_{s+m} = p_m$

Concrete examples:

- The pattern $P = \text{ell}$ appears in the text $T = \text{Hello, world}$ with a shift 1.
- The pattern $P = \text{Helo}$ does not appear in the text T .

Applications: Finding a pattern in a text

Computer science page on wikipedia (first two paragraphs)

Computer science is the study of processes that interact with data and that can be represented as data in the form of programs. It enables the use of algorithms to manipulate, store, and communicate digital information. A computer scientist studies the theory of computation and the practice of designing software systems.

Its fields can be divided into theoretical and practical disciplines. Computational complexity theory is highly abstract, while computer graphics emphasizes real-world applications. Programming language theory considers approaches to the description of computational processes, while computer programming itself involves the use of programming languages and complex systems. Human-computer interaction considers the challenges in making computers useful, usable, and accessible.

How much time does the **pattern** "computer" appears in these two paragraphs?

Applications: Finding a pattern in a text

Computer science page on wikipedia (first two paragraphs)

Computer science is the study of processes that interact with data and that can be represented as data in the form of programs. It enables the use of algorithms to manipulate, store, and communicate digital information. A **computer** scientist studies the theory of computation and the practice of designing software systems.

Its fields can be divided into theoretical and practical disciplines. Computational complexity theory is highly abstract, while **computer** graphics emphasizes real-world applications. Programming language theory considers approaches to the description of computational processes, while **computer** programming itself involves the use of programming languages and complex systems. Human-**computer** interaction considers the challenges in making **computers** useful, usable, and accessible.

How much time does the **pattern** "computer" appears in these two paragraphs?

Find a pattern in a DNA sequence:

- $T = \text{GTGCTATGCTGATGCTGACTTATATGCTACGTTTCGGCTATC}$

How much time does the **pattern** GCTA appears in T?

Find a pattern in a DNA sequence:

- $T = \text{GT}\textcolor{red}{\text{GCTA}}\text{TGCTGATGCTGACTTATAT}\textcolor{red}{\text{GCTA}}\text{CGTTCG}\textcolor{red}{\text{GCTA}}\text{TC}$

How much time does the **pattern** $\textcolor{red}{\text{GCTA}}$ appears in T ?

Applications: Web page relevance to queries

String matching can also be used as a first preprocessing step to provide pages relevant to queries.

Google

UCL

All

Images

Maps

News

Videos

More

Settings

Tools

About 53,300,000 results (0.61 seconds)

UCL - London's Global University

<https://www.ucl.ac.uk/>

UCL (University College London) is London's leading multidisciplinary university, with 11000 staff, 35000 students and an annual income of over £1bn.

Results from ucl.ac.uk

Undergraduate

Degrees - Subject areas - Entry requirements - How to apply - ...

UCL Graduate degrees

Taught degrees - Research degrees - Downloads - ...

Prospective students

Taught degrees - Degrees - UCL Graduate degrees - Apply - ...

International Students

Information for international students interested in studying ...

Student Accommodation

Undergraduates - Postgraduates - Catered Accommodation - ...

UCL Press

About UCL Press - Browse Books and Journals - Open Access - ...

People also ask

Is UCL prestigious?

What is the meaning of UCL?

What is the acceptance rate for UCL?

Is UCL a top university?

Feedback


UCL (@ucl) · Twitter

<https://twitter.com/ucl>


The Bartlett, UCL's Faculty of the Built Environment, is turning 100 in 2019, bringing you 100 stories over 100 days. Join in the celebration and share your stories of what The Bartlett means to you using #Bartlett100.

Final-year undergraduates can help their department earn funds to spend on students by completing their NSS questionnaire. Many departments use the funds to help students celebrate graduating! bit.ly/2MP8j1

Happy Chinese New Year! Wherever you're celebrating we hope you have a great day. #CHN2019 #YearOfThePig pic.twitter.com/qzE1PH...



See photos



See outside

University College de Londres

Site Web

Résumé

Enregistrer

Université à Londres, Angleterre

L'University College London, couramment abrégé « UCL », est la plus ancienne université de l'université de Londres. [Wikipedia](#)

Adresse : Gower St, Bloomsbury, London WC1E 6BT, UK

Recteur : Michael Arthur

Devise en anglais : "Let all come who by merit deserve the most reward"


Budget : £1 431.7 billion (university); £1 451.1 billion (consolidated) (2017-18)

Le savez-vous ? : University College London is the third-largest university in the United Kingdom by enrolment (37,905 total students). [wikipedia.org](#)


[Suggérer une modification](#)

Anciens élèves célèbres


Voir d'autres éléments (plus de 45)




Alexander Graham Bell




Mohandas Karamchand Gandhi



Chris Martin




Christopher Nolan



Ricky Gervais

Profiles



Instagram

4

Applications: Web page relevance to queries

String matching can also be used as a first preprocessing step to provide pages relevant to queries.

Google

UCL

All

Images

Maps

News

Videos

More

Settings

Tools

About 53,300,000 results (0.61 seconds)

UCL - London's Global University
<https://www.ucl.ac.uk/> ▼
UCL (University College London) is London's leading multidisciplinary university, with 11000 staff, 35000 students and an annual income of over £1bn.

Results from ucl.ac.uk

Undergraduate
Degrees - Subject areas - Entry requirements - How to apply - ...

UCL Graduate degrees
Taught degrees - Research degrees - Downloads - ...

Prospective students
Taught degrees - Degrees - UCL Graduate degrees - Apply - ...

International Students
Information for international students interested in studying ...

Student Accommodation
Undergraduates - Postgraduates - Catered Accommodation - ...

UCL Press
About UCL Press - Browse Books and Journals - Open Access - ...

People also ask

Is UCL prestigious?

What is the meaning of UCL?

What is the acceptance rate for UCL?

Is UCL a top university?



Feedback

UCL (@ucl) · Twitter
<https://twitter.com/ucl>

The Bartlett, UCL's Faculty of the Built Environment, is turning 100 in 2019, bringing you 100 stories over 100 days. Join in the celebration and share your stories of what The Bartlett means to you using #Bartlett100.

Final-year undergraduates can help their department earn funds to spend on students by completing their NSS questionnaire. Many departments use the funds to help students celebrate graduating! bit.ly/2MP8j1

Happy Chinese New Year! Wherever you're celebrating we hope you have a great day. #CHN2019 #YearOfThePig pic.twitter.com/qzE1PH...



See photos

See outside

University College de Londres

Site Web

Réserver

Enregistrer

Université à Londres, Angleterre

L'University College London, couramment abrégé « UCL », est la plus ancienne université de l'université de Londres. [Wikipedia](#)

Adresse : Gower St, Bloomsbury, London WC1E 6BT, UK

Recteur : Michael Arthur


Devise en anglais : "Let all come who by merit deserve the most reward"

Budget : £1.431.7 billion (university); £1.451.1 billion (consolidated) (2017-18)


Le savez-vous ? : University College London is the third-largest university in the United Kingdom by enrolment (37,905 total students). [wikipedia.org](#)

[Suggérer une modification](#)


Anciens élèves célèbres




Alexander Graham Bell




Mohandas Karamchand Gandhi



Chris Martin




Christopher Nolan



Ricky Gervais

Profiles



Instagram

4

Applications: Web page relevance to queries

String matching can also be used as a first preprocessing step to provide pages relevant to queries.

The screenshot shows a Google search for 'UCL'. The search bar at the top contains 'UCL' and shows 'About 53,300,000 results (0.61 seconds)'. Below the search bar, there are tabs for 'All', 'Images', 'Maps', 'News', 'Videos', 'More', 'Settings', and 'Tools'. The 'All' tab is selected.

The search results are displayed in a grid. The first result is 'UCL - London's Global University' with the URL 'https://www.ucl.ac.uk/'. Below this, there is a description: 'UCL (University College London) is London's leading multidisciplinary university, with 11000 staff, 35000 students and an annual income of over £1bn.' Below the description, there is a search bar with 'Results from ucl.ac.uk' and a magnifying glass icon.

Below the search bar, there are four columns of links:

- Undergraduate**
Degrees - Subject areas - Entry requirements - How to apply - ...
- UCL Graduate degrees**
Taught degrees - Research degrees - Downloads - ...
- Prospective students**
Taught degrees - Degrees - UCL Graduate degrees - Apply - ...
- International Students**
Information for international students interested in studying ...
- Student Accommodation**
Undergraduates - Postgraduates - Catered Accommodation - ...
- UCL Press**
About UCL Press - Browse Books and Journals - Open Access - ...

Below the links, there is a section 'People also ask' with four questions:

- Is UCL prestigious?
- What is the meaning of UCL?
- What is the acceptance rate for UCL?
- Is UCL a top university?

Below the 'People also ask' section, there is a section 'UCL (@ucl) - Twitter' with three tweets:

- The Bartlett, UCL's Faculty of the Built Environment, is turning 100 in 2019, bringing you 100 stories over 100 days. Join in the celebration and share your stories of what The Bartlett means to you using #Bartlett100.
- Final-year undergraduates can help their department earn funds to spend on students by completing their NSS questionnaire. Many departments use the funds to help students celebrate graduation! bit.ly/2MPH11
- Happy Chinese New Year! Wherever you're celebrating we hope you have a great day. #CHN2019 #YearOfThePig pic.twitter.com/qzE1PH...

On the right side of the search results, there is a map of UCL and a section 'University College de Londres' with links to 'Site Web', 'Règlement', and 'Enregistrer'. Below this, there is a section 'Anciens élèves célèbres' with six portraits of famous alumni: Alexander Graham Bell, Mohandas Karamchand Gandhi, Chris Martin, Christopher Nolan, and Ricky Gervais.

Naive solution to string matching

Building a naive algorithm

How much time and where does the pattern *AABA* appears in the text?

- Text: $T = AABAACAADAABAABA$
- Pattern: $P = AABA$

Building a naive algorithm

How much time and where does the pattern AABA appears in the text?

- Text: $T = AABAACAADAABAABA$
- Pattern: $P = AABA$

Shift	0
T	A A B A A C A A D A A B A A B A
P	A A B A

Building a naive algorithm

How much time and where does the pattern AABA appears in the text?

- Text: $T = AABAACAADAABAABA$
- Pattern: $P = AABA$

Shift	1															
T	A	A	B	A	A	C	A	A	D	A	A	B	A	A	B	A
P		A	A	B	A											

Building a naive algorithm

How much time and where does the pattern AABA appears in the text?

- Text: $T = AABAACAADAABAABA$
- Pattern: $P = AABA$

Shift	2															
T	A	A	B	A	A	C	A	A	D	A	A	B	A	A	B	A
P			A	A	B	A										

Building a naive algorithm

How much time and where does the pattern AABA appears in the text?

- Text: $T = AABAACAADAABAABA$
- Pattern: $P = AABA$

Shift	3															
T	A	A	B	A	A	C	A	A	D	A	A	B	A	A	B	A
P				A	A	B	A									

Building a naive algorithm

How much time and where does the pattern AABA appears in the text?

- Text: $T = AABAACAADAABAABA$
- Pattern: $P = AABA$

Shift	0									9				12				
T	A	A	B	A	A	C	A	A	D	A	A	B	A	A	B	A		
P	A	A	B	A						A	A	B	A	A	B	A		

P occurs in T at shifts 0, 9 and 12.

Naive algorithm

```
1: NAIVE-STRING-MATCHING( $T, P$ ):
2:    $n = \text{length}(T)$ 
3:    $m = \text{length}(P)$ 
4:   for  $i$  in  $[0, n - m + 1]$  do
5:     for  $j$  in  $[1, m]$  do
6:       if  $T[i + j] \neq P[j]$  then
7:         break
8:       end if
9:     end for
10:    if  $j == m$  then
11:      print "Pattern found at shift ",  $i$ 
12:    end if
13:  end for
```

Naive algorithm

```
1: NAIVE-STRING-MATCHING( $T, P$ ):
2:    $n = \text{length}(T)$ 
3:    $m = \text{length}(P)$ 
4:   for  $i$  in  $[0, n - m + 1]$  do
5:     for  $j$  in  $[1, m]$  do
6:       if  $T[i + j] \neq P[j]$  then
7:         break
8:       end if
9:     end for
10:    if  $j == m$  then
11:      print "Pattern found at shift ",  $i$ 
12:    end if
13:  end for
```

Naive algorithm

```
1: NAIVE-STRING-MATCHING( $T, P$ ):
2:    $n = \text{length}(T)$ 
3:    $m = \text{length}(P)$ 
4:   for  $i$  in  $[0, n - m + 1]$  do
5:     for  $j$  in  $[1, m]$  do
6:       if  $T[i + j] \neq P[j]$  then
7:         break
8:       end if
9:     end for
10:    if  $j == m$  then
11:      print "Pattern found at shift ",  $i$ 
12:    end if
13:  end for
```

Naive algorithm

```
1: NAIVE-STRING-MATCHING( $T, P$ ):
2:    $n = \text{length}(T)$ 
3:    $m = \text{length}(P)$ 
4:   for  $i$  in  $[0, n - m + 1]$  do
5:     for  $j$  in  $[1, m]$  do
6:       if  $T[i + j] \neq P[j]$  then
7:         break
8:       end if
9:     end for
10:    if  $j == m$  then
11:      print "Pattern found at shift ",  $i$ 
12:    end if
13:  end for
```


Naive algorithm complexity

- **Best case scenario:** the first character of the pattern does not appear in the text:
 - $T = AABABABABABABABABA$
 - $P = ZABA$
 - Best case **complexity:** $\mathcal{O}(m)$

Naive algorithm complexity

- **Best case scenario:** the first character of the pattern does not appear in the text:
 - $T = AABABABABABABABABA$
 - $P = ZABA$
 - Best case **complexity:** $\mathcal{O}(m)$
- **Worst case scenario:** either all characters of the pattern match the full text:
 - $T = AAAAAAAAAAAAAAAAAAAAAA$
 - $P = AAA$

Naive algorithm complexity

- **Best case scenario:** the first character of the pattern does not appear in the text:
 - $T = AABABABABABABABABA$
 - $P = ZABA$
 - Best case **complexity:** $\mathcal{O}(m)$
- **Worst case scenario:** either all characters of the pattern match the full text:
 - $T = AAAAAAAAAAAAAAAAAAAAAA$
 - $P = AAA$or only the last characters of both strings are the same:
 - $T = AAAAAAAAAAAAAAAAAAAAZ$
 - $P = AAZ$

Worst case **complexity:** $\mathcal{O}(m(n - m))$

What is the problem with the naive algorithm?

Improving over the naive algorithm

What is the problem with the naive algorithm?

Shift	0															
T	A	A	B	A	A	C	A	A	D	A	A	B	A	A	B	A
P	A	A	B	A												

Improving over the naive algorithm

What is the problem with the naive algorithm?

Shift	1															
T	A	A	B	A	A	C	A	A	D	A	A	B	A	A	B	A
P		A	A	B	A											

Improving over the naive algorithm

What is the problem with the naive algorithm?

Shift	2															
T	A	A	B	A	A	C	A	A	D	A	A	B	A	A	B	A
P			A	A	B	A										

Improving over the naive algorithm

What is the problem with the naive algorithm?

Shift	3															
T	A	A	B	A	A	C	A	A	D	A	A	B	A	A	B	A
P				A	A	B	A									

Improving over the naive algorithm

What is the problem with the naive algorithm?

Shift	4															
T	A	A	B	A	A	∅	A	A	D	A	A	B	A	A	B	A
P					A	A	B	A								

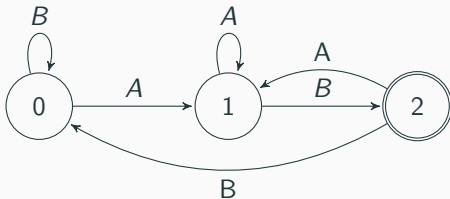
At each shift, we scan again the full substring until we find a character that does not match!

String matching with finite automaton

Finite automata definition

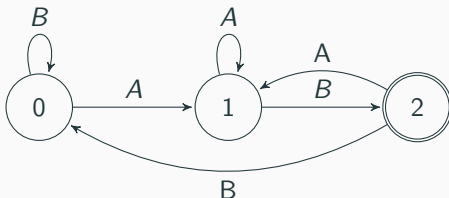
- Finite automata $M = (\mathcal{Q}, q_0, A, \Sigma, \delta)$:
 - \mathcal{Q} is a finite set of **states**,
 - $q_0 \in \mathcal{Q}$ is the **start state**,
 - $A \subset \mathcal{Q}$ is the set of **accepting states**,
 - Σ is a finite **input alphabet**,
 - $\delta : (\mathcal{Q}, \Sigma) \rightarrow \mathcal{Q}$, called the **transition function of M** : determines the future state based on the current state and the current input symbol.

Example: a simple automata that matches AB



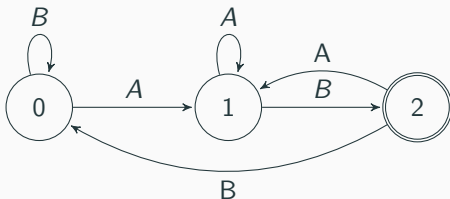
- **Starting state:** 0
- **States:** 0, 1 and 2
- **Accepting state:** 2
- **Alphabet/vocabulary:**
 $\Sigma = \{A, B\}$

Example: a simple automata that matches AB



- **Starting state:** 0
- **States:** 0, 1 and 2
- **Accepting state:** 2
- **Alphabet/vocabulary:**
 $\Sigma = \{A, B\}$
- The edges of an automata allow to move from one state to another one by "consuming" a character. E.g. we move from state 0 to 1 "consuming" an *A* and from state 1 to 2 "consuming" a *B*.
- Whenever the automata arrives to an accepting state, it means we just found the pattern *P* in text *T*.

Example: a simple automata that matches AB



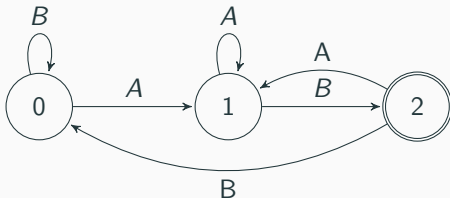
Equivalent representation of the automata:

$$\begin{array}{c} A \quad B \\ \begin{matrix} 0 \\ 1 \\ 2 \end{matrix} \begin{pmatrix} 1 & 0 \\ 1 & 2 \\ 1 & 0 \end{pmatrix} \end{array}$$

Example on a simple text:

- $T = ABAABBB$
- $P = AB$

Example: a simple automata that matches AB



Equivalent representation of the automata:

	A	B
0	1	0
1	1	2
2	1	0

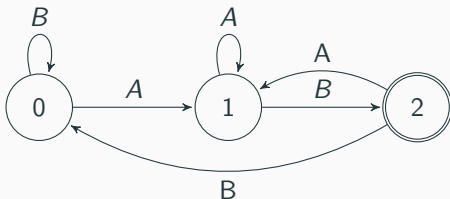
Example on a simple text:

- $T = ABAABBB$
- $P = AB$

Execution:

- Goes from state 0 to 1 "consuming" $t_1 = A$

Example: a simple automata that matches AB



Equivalent representation of the automata:

	A	B
0	1	0
1	1	2
2	1	0

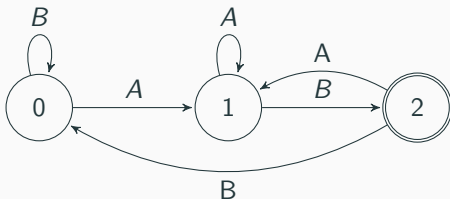
Example on a simple text:

- $T = ABAABBB$
- $P = AB$

Execution:

- Goes from state 1 to 2 "consuming"
 $t_2 = B$, 1st match at shift $s = 0$

Example: a simple automata that matches AB



Equivalent representation of the automata:

	A	B
0	1	0
1	1	2
2	1	0

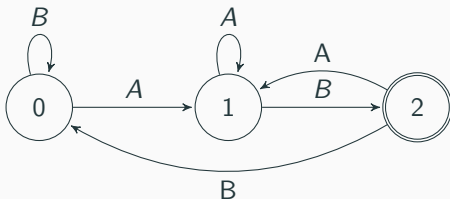
Example on a simple text:

- $T = ABAABBB$
- $P = AB$

Execution:

- Goes from state 2 to 1 "consuming" $t_3 = A$

Example: a simple automata that matches AB



Equivalent representation of the automata:

	A	B
0	1	0
1	1	2
2	1	0

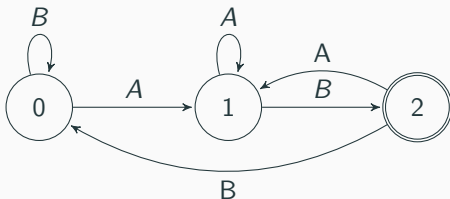
Example on a simple text:

- $T = ABAABBB$
- $P = AB$

Execution:

- Goes from state 1 to 1 "consuming" $t_4 = A$

Example: a simple automata that matches AB



Equivalent representation of the automata:

	A	B
0	1	0
1	1	2
2	1	0

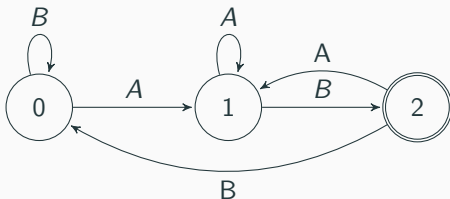
Example on a simple text:

- $T = ABAABBB$
- $P = AB$

Execution:

- Goes from state 1 to 2 "consuming"
 $t_5 = B$, 2nd match at shift $s = 3$

Example: a simple automata that matches AB



Equivalent representation of the automata:

	A	B
0	1	0
1	1	2
2	1	0

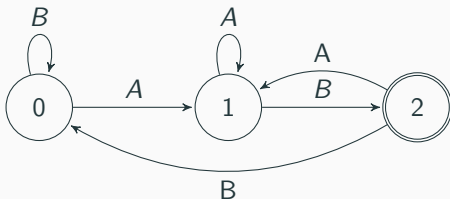
Example on a simple text:

- $T = ABAABBB$
- $P = AB$

Execution:

- Goes from state 2 to 0 "consuming" $t_6 = B$

Example: a simple automata that matches AB



Equivalent representation of the automata:

	A	B
0	1	0
1	1	2
2	1	0

Example on a simple text:

- $T = ABAABBB$
- $P = AB$

Execution:

- Goes from state 0 to 0 "consuming" $t_7 = B$

```
1: FA-STRING-MATCHER( $T$ , next-state,  $m$ ):
2:    $n = \text{length}(T)$ 
3:   state = 0
4:   for  $i$  in  $[1, n]$  do
5:     state = next-state[state,  $t_i$ ]
6:     if state == accepting state then
7:       print "The pattern occurs with shift",  $i - m$ 
8:     end if
9:   end for
```

Finite automata string matcher

```
1: FA-STRING-MATCHER( $T$ , next-state,  $m$ ):
2:    $n = \text{length}(T)$ 
3:    $\text{state} = 0$ 
4:   for  $i$  in  $[1, n]$  do
5:      $\text{state} = \text{next-state}[\text{state}, t_i]$ 
6:     if  $\text{state} == \text{accepting state}$  then
7:       print "The pattern occurs with shift",  $i - m$ 
8:     end if
9:   end for
```

Finite automata string matcher

```
1: FA-STRING-MATCHER( $T$ , next-state,  $m$ ):
2:    $n = \text{length}(T)$ 
3:   state = 0
4:   for  $i$  in  $[1, n]$  do
5:     state = next-state[state,  $t_i$ ]
6:     if state == accepting state then
7:       print "The pattern occurs with shift",  $i - m$ 
8:     end if
9:   end for
```


Finite automata string matcher

```
1: FA-STRING-MATCHER( $T$ , next-state,  $m$ ):  
2:    $n = \text{length}(T)$   
3:   state = 0  
4:   for  $i$  in  $[1, n]$  do  
5:     state = next-state[state,  $t_i$ ]  
6:     if state == accepting state then  
7:       print "The pattern occurs with shift",  $i - m$   
8:     end if  
9:   end for
```

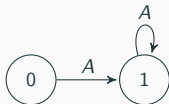
Building a finite automata for $P = ACACAGA$

- $\Sigma = \{A, T, C, G\}$
- $P = ACACAGA$
- **Automata matching:**



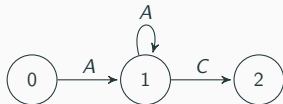
Building a finite automata for $P = ACACAGA$

- $\Sigma = \{A, T, C, G\}$
- $P = ACACAGA$
- **Automata matching: A**



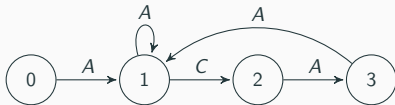
Building a finite automata for $P = ACACAGA$

- $\Sigma = \{A, T, C, G\}$
- $P = ACACAGA$
- **Automata matching: AC**



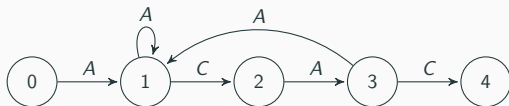
Building a finite automata for $P = ACACAGA$

- $\Sigma = \{A, T, C, G\}$
- $P = ACACAGA$
- **Automata matching: ACA**



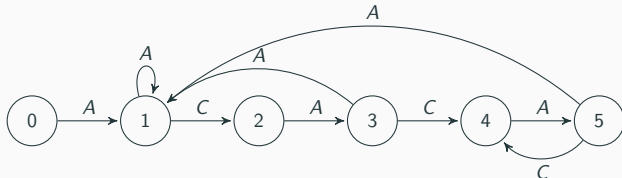
Building a finite automata for $P = ACACAGA$

- $\Sigma = \{A, T, C, G\}$
- $P = ACACAGA$
- **Automata matching: ACAC**



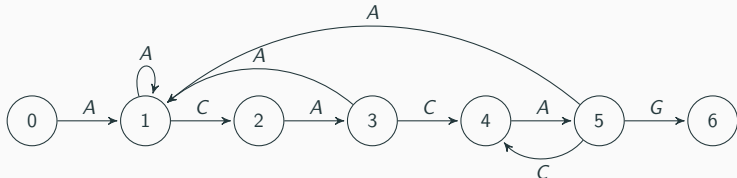
Building a finite automata for $P = ACACAGA$

- $\Sigma = \{A, T, C, G\}$
- $P = ACACAGA$
- **Automata matching: ACACA**



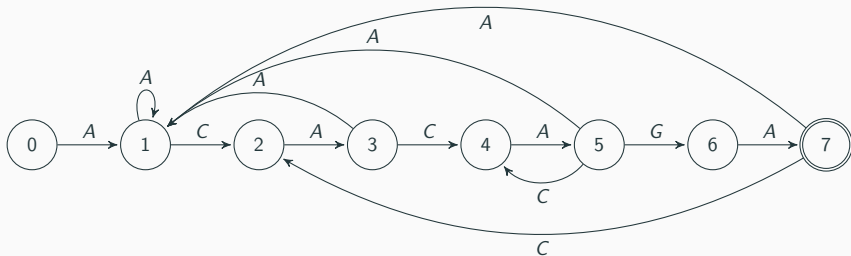
Building a finite automata for $P = ACACAGA$

- $\Sigma = \{A, T, C, G\}$
- $P = ACACAGA$
- **Automata matching: ACACAG**

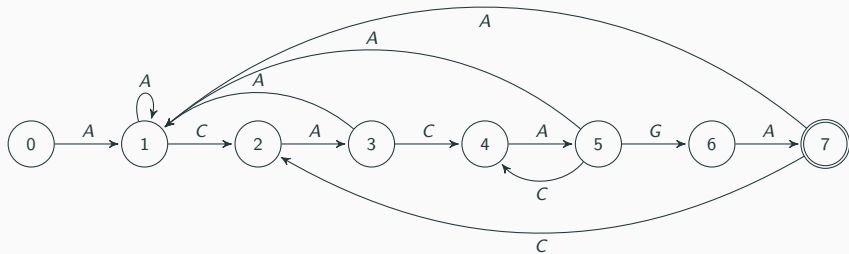


Building a finite automata for $P = ACACAGA$

- $\Sigma = \{A, T, C, G\}$
- $P = ACACAGA$
- Automata matching: **ACACAGA**



Building a finite automata for $P = ACACAGA$



	A	T	C	G
0	1	0	0	0
1	1	0	2	0
2	3	0	0	0
3	1	0	4	0
4	5	0	0	0
5	1	0	4	6
6	7	0	0	0
7	1	0	2	0

Building the next-state array automatically

Ideas:

1. We need to get the next state from the current state for every possible character.
2. For a character c and state s we compute the longest prefix of the pattern P that is also a suffix of $P[0, s - 1]|c$.

Where:

- $P[1, s]|c$ means that we concatenate c to the string $P[1, s]$. E.g. if $P = ABC$ and $c = D$, $P[1, 3]|c = ABCD$.
- A **prefix** of a string $S = s_1, s_2, \dots, s_n$ is a string $p = s_1, s_2, \dots, s_k, k \leq n$.
E.g. if $S = ABCDEF$ then $p = ABC$ is a prefix of S .
- A **suffix** of a string $S = s_1, s_2, \dots, s_n$ is a string $p = s_k, \dots, s_{n-1}, s_n, k \geq 1$.
E.g. if $S = ABCDEF$ then $p = EF$ is a suffix of S .

Building the next-state array automatically

```
1: NEXT-STATE-ARRAY( $P$ ):  
2:    $m = \text{length}(P)$   
3:   for state in  $[0, m]$  do  
4:     for char in  $\Sigma$  do  
5:        $i = \min(m, \text{state} + 1)$   
6:       while  $P[1, i]$  is not a suffix of pattern  $P[1, \text{state}]||\text{char}$  do  
7:          $i = i - 1$   
8:       end while  
9:        $\delta(\text{state}, \text{char}) = i$   
10:    end for  
11:  end for  
12:  return  $\delta$ 
```

Complexity of NEXT-STATE-ARRAY: $\mathcal{O}(m^3 * |\Sigma|)$