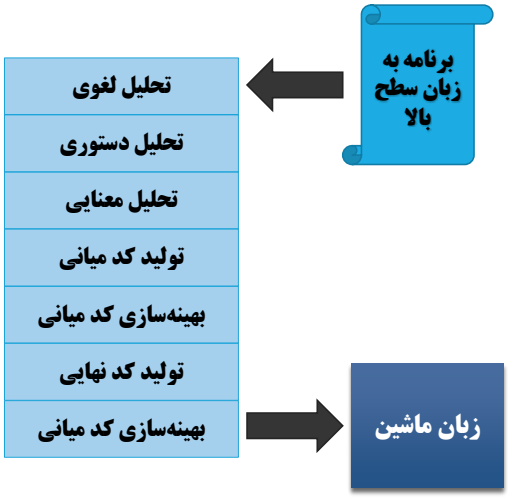
تحلیل لغوی

فصل یک: تحلیل لغوی

اصول طراحی کامپایلرها

دکتر امیر جلالی

Compiler



مراجع این فصل

- کتاب اژدها
  - فصل سوم
  - ۳-۱ الی ۳-۷
  - این فصل از کتاب جزییات زیادی در ارتباط با کد کامپایلر به زبان C دارد، که نیازی به مطالعه آنها نیست.
- مباحث مربوط به زبان‌های منظم را از منابع درس نظریه نیز می‌توانید مطالعه کنید.
  - نظریه زبان‌ها و ماشین‌ها (پیتر لینز)، فصل اول: زبانهای منظم

فهرست

تحلیل لغوی

متن برنامه

- متن ورودی برنامه از دید ما

```
if (i == j)
    Z = 0;
else
    Z = 1;
```

- متن ورودی برنامه از دید کامپیوتر (دنباله‌ای از کاراکترها)

```
\tif (i == j)\n\t\tz = 0;\n\telse\n\t\tz = 1;
```

تحلیل‌گر لغوی

Scanning a Source File

w	h	i	l	e		(	1	3	7		<		i	)	\n	\t	+	+	i	;

Scanning a Source File

Scanning a Source File

Scanning a Source File

Scanning a Source File

Scanning a Source File

Scanning a Source File

Scanning a Source File

T_While

Scanning a Source File

این دنباله‌ی کاراکترها از متن ورودی، **لغت** نامیده می‌شوند.

T_While

این **توکن** نامیده می‌شود. می‌توانیم آن را نوع لغت خوانده شده بنامیم.

Scanning a Source File

T_While

Scanning a Source File

T_While

Scanning a Source File

T_While

بعضی از توکن‌ها دور ریخته می‌شوند، چرا که در بخش‌های بعدی کامپایلر استفاده‌ای ندارد. مانند فاصله و خط جدید (new line).

Scanning a Source File

T_While

Scanning a Source File

T_While

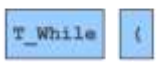
Scanning a Source File

T_While

پاییز 95 اصول طراحی کامپایلر، امیر جلالی بیدگلی

Scanning a Source File

Scanning a Source File



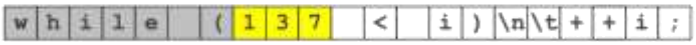
پاییز 95 اصول طراحی کامپایلر، امیر جلالی بیدگلی

Scanning a Source File

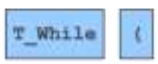
Scanning a Source File

پاییز 95 اصول طراحی کامپایلر، امیر جلالی بیدگلی

Scanning a Source File



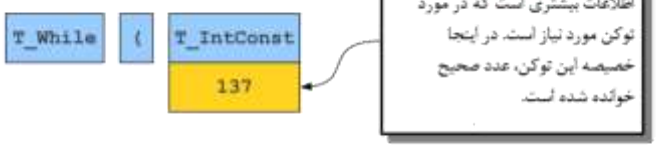
Scanning a Source File



پاییز 95 اصول طراحی کامپایلر، امیر جلالی بیدگلی

Scanning a Source File

Scanning a Source File



پاییز 95 اصول طراحی کامپایلر، امیر جلالی بیدگلی

توکن، لغت و الگو

- توکن: کوچکترین واحد معنادار در کامپایل
  - مجموعه‌ای از لغات را مشخص می‌کند.
  - هر توکن شامل نوع لغت و خصیصه در صورت نیاز است.
    - خصیصه اطلاعات بیشتری در مورد توکن در اختیار می‌دهد
      - مثال: مقدار عدد در اعداد، اشاره‌گر به جدول نمادها در شناسه
    - معمولا وقتی توکن فقط یک لغت را شامل می‌شود، خصیصه لازم نیست.
      - مانند توکن‌های کلمات کلیدی (While)
- لغت: دنباله‌ای از کاراکترها که یک توکن را تشکیل می‌دهد.
- الگو: توصیف لغاتی که عضو هر توکن هستند.

نمونه‌های از توکن‌ها

توکن	الگو	نمونه لغت
If	I + f	If
Else	E + l + s + e	Else
Comparison	<, >, =>, =<, ==, !=	== =>
Number	رشته‌هایی که فقط با ۰ تا ۹ ساخته شده باشند	120, 35
Literal	هر عبارتی بین دو «»	“asasa” “a b”

توکن‌های متداول در زبان‌ها

- کلمات کلیدی: یک توکن به ازای هر کلمه کلیدی
- عملگرها: یک توکن به ازای هر عملگر یا مجموعه عملگرهای مشابه
  - مانند عملگرهای مقایسه‌ای Comparison
- ثابت‌ها: یک یا چند توکن به ازای ثابت‌های رشته‌ای، عددی، ...
- جداکننده‌ها: یک تکون به ازای هر جداکننده
  - مانند (،)، ؛، ;

برخی چالش‌ها

مشکلات تحلیل‌گر لغوی (۱)

- در برخی زبان‌ها کلمات کلیدی رزرو شده نیستند.
  - مانند PL/1

مشکلات تحلیل‌گر لغوی (۱)

- در برخی زبان‌ها کلمات کلیدی رزرو شده نیستند.
  - مانند PL/1

```
IF THEN THEN THEN = ELSE; ELSE ELSE = IF
```

مشکلات تحلیل‌گر لغوی (۱)

- در برخی زبان‌ها کلمات کلیدی رزرو شده نیستند.
  - مانند PL/1: کلمات کلیدی می‌توانند به عنوان شناسه نیز استفاده شوند.

```
IF THEN THEN THEN = ELSE; ELSE ELSE = IF
```

مشکلات تحلیل‌گر لغوی (۲)

- در برخی از زبان‌ها، فاصله‌های خالی نادیده گرفته می‌شوند.
  - مانند FORTRAN, Algol68
  - VAR1 دقیقا معادل است با VA R 1

برگرفته از مطالب Keith Schwarz (Standford)

مشکلات تحلیل‌گر لغوی (۲)

- در برخی از زبان‌ها، فاصله‌های خالی نادیده گرفته می‌شوند.
  - مانند FORTRAN, Algol68
  - VAR1 دقیقا معادل است با VA R 1

```
DO 5 I = 1,25
DO 5 I = 1.25
```

برگرفته از مطالب Keith Schwarz (Standford)

مشکلات تحلیل‌گر لغوی (۲)

- در برخی از زبان‌ها، فاصله‌های خالی نادیده گرفته می‌شوند.
  - مانند FORTRAN, Algol68
  - VAR1 دقیقا معادل است با VA R 1

```
DO 5 I = 1,25
DO5I   = 1.25
```

برگرفته از مطالب Keith Schwarz (Standford)

برخی دیگر از مشکلات!

I am having some difficulty compiling a C++ program that I've written.

This program is very simple and, to the best of my knowledge, conforms to all the rules set forth in the C++ Standard. [...]

The program is as follows:

Source:
http://stackoverflow.com/questions/5508110/why-is-this-program-erroneously-rejected-by-three-c-compilers

برگرفته از مطالب Keith Schwarz (Standford)

برخی دیگر از مشکلات!

I am having some difficulty compiling a C++ program that I've written.

This program is very simple and, to the best of my knowledge, conforms to all the rules set forth in the C++ Standard. [...]

The program is as follows:

```
#include <iostream>
int main(int argc, char** argv)
{
    std::cout << "Hello world!" << std::endl;
    return 0;
}
```

Source:
http://stackoverflow.com/questions/5508110/why-is-this-program-erroneously-rejected-by-three-c-compilers

برگرفته از مطالب Keith Schwarz (Standford)

برخی دیگر از مشکلات!

I am having some difficulty compiling a C++ program that I've written.

This program is very simple and, to the best of my knowledge, conforms to all the rules set forth in the C++ Standard. [...]

The program is as follows:

```
#include <iostream>

int main(int argc, char** argv)
{
    std::cout << "Hello world!" << std::endl;
    return 0;
}
```

```
> g++ helloworld.png
helloworld.png: file not recognized: File format not recognized
collect2: ld returned 1 exit status
```

Source:
http://stackoverflow.com/questions/5508110/why-is-this-program-erroneously-rejected-by-three-c-compilers

برگرفته از مطالب Keith Schwarz (Standford)

...

جمع‌بندی

- پویشگر: متن برنامه را از چپ به راست بخواند:
  - متن ورودی را به لغات بخش کند (lexeme)
  - توکن مربوط به هر لغت را تشخیص دهد.
- گاهی لازم است لغات پیش رو مکان نما را هم بخواند.
  - Lookahead
  - برای حل مشکلاتی مانند زبان‌های فرترن و PL/1

مراحل انجام تحلیل‌گر لغوی

برگرفته از مطالب Keith Schwarz (Standford)

- الگو توکن‌ها را توصیف کنیم؟
  - روش توصیف: عبارت‌های منظم
- کاراکترهای خوانده‌شده را با الگوها تطبیق دهیم؟
  - مکانیزم پیاده‌سازی شناسایی الگوها: ماشین حالت

عبارت‌های منظم

عبارات منظم

- عبارات منظم: هر عبارتی که بتوانیم با ترکیب حروف الفبا و عملگرهای زیر بسازیم (به ترتیب اولویت)
  - * : بستار ستاره
    - رشته‌هایی که با صفر بار تکرار یا بیشتر می‌توان ساخت
  - . : الحاق
    - به هم چسباندن دو رشته
  - | : عملگر یا
    - انتخاب یکیاز دو رشته

مثال

a.(b | C)

- رشته‌های ab یا ac

مثال

a.(a | b | c)*

- تمام رشته‌هایی با a شروع می‌شوند.

مثال

(0 | 1)*00(0 | 1)*

- فرض کنید، حروف الفبا 0 و 1 باشند،
- عبارت منظم تمام رشته‌هایی که حتما دارای زیررشته 00 هستند؟

تعریف رسمی عبارت‌های منظم

- پایه:
  - ε یک عبارت منظم است. $L(\varepsilon) = \{\varepsilon\}$
  - a (یکی از حروف الفبا) یک عبارت منظم است. $L(a) = \{a\}$
- استقرا: اگر R_1 و R_2 دو عبارت منظم باشند، آنگاه:
  - R_1^* یک عبارت منظم است. $\mathrm{L}(R_1^*) = L(R_1)^*$
  - $R_1.R_2$ یک عبارت منظم است. $\mathrm{L}(R_1.R_2) = L(R_1).L(R_2)$
  - $R_1|R_2$ یک عبارت منظم است. $\mathrm{L}(R_1|R_2) = L(R_1) \cup L(R_2)$

- فرض کنید الفبا فقط 0 و1 باشد.
- عبارت منظم اعداد چهار رقمی؟

$$(0 \mid 1)(0 \mid 1)(0 \mid 1)(0 \mid 1)$$

- می‌توان به صورت زیر خلاصه نوشت:

$$(0|1)^4$$

- فرض کنید الفبا فقط 0 و1 باشد.
- عبارت منظم رشته‌هایی که حداکثر یک صفر دارند؟

$$1^*(0|\varepsilon)1^*$$

- می‌توان به صورت زیر خلاصه نوشت:

$$1^*0?\,1^*$$

- فرض کنید الفبا فقط a, b, c, d باشد.
- عبارت منظم رشته‌هایی که با a شروع و تمام می‌شوند؟

$$a(a|b|c|d)^*a$$

- می‌توان به صورت زیر خلاصه نوشت:

$$a[a-d]^*a$$

- عبارت منظم برای ای میل‌های معتبر (فقط با حروف و نقطه)؟

$$[a-z][a-z]^*('.'[a-z][a-z]^*)*@$$
$$[a-z][a-z]^*'.'[a$$

- عبارت منظم اعداد صحیح زوج؟

$$(+|-)[0-9]^*[02468]$$

نمونه‌های از توکن‌ها

توکن	الگو	نمونه لغت
If	If	If
Else	Else	Else
Comparison	(< | > | <= | => | == | !=)	== =>
Number	[0-9]+	120, 35
Literal	"Σ * "	“asasa” “a b”

سوال؟

- عبارت منظم برای select در SQL؟

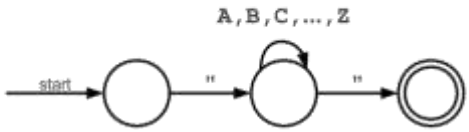
سوال؟

- عبارت منظم برای comment های به شکل زیر؟
  - /* ..... */

روش پیاده‌سازی

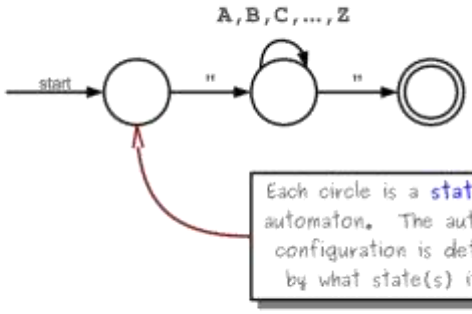
ماشین حالت

A Simple Automaton



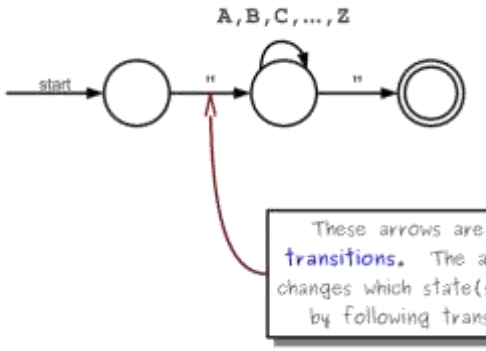
برگرفته از مطالب Keith Schwarz (Standford)

A Simple Automaton



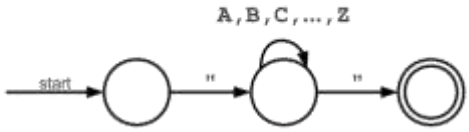
برگرفته از مطالب Keith Schwarz (Standford)

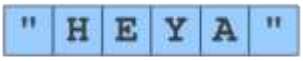
A Simple Automaton



برگرفته از مطالب Keith Schwarz (Standford)

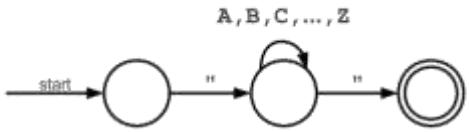
A Simple Automaton





برگرفته از مطالب Keith Schwarz (Standford)

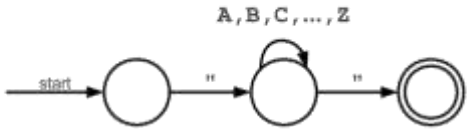
A Simple Automaton

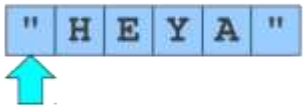


The automaton takes a string as input and decides whether to accept or reject the string.

برگرفته از مطالب Keith Schwarz (Standford)

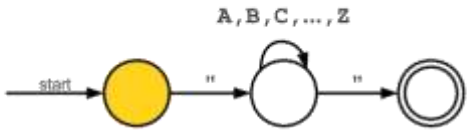
A Simple Automaton

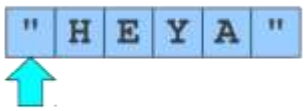




برگرفته از مطالب Keith Schwarz (Standford)

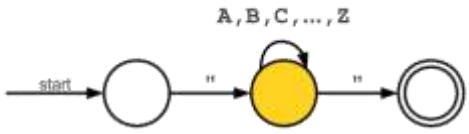
A Simple Automaton

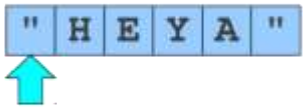




برگرفته از مطالب Keith Schwarz (Standford)

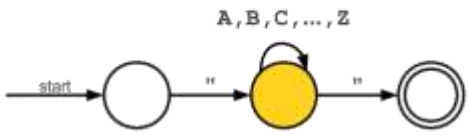
A Simple Automaton

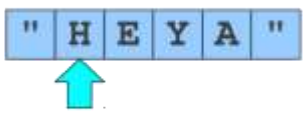




برگرفته از مطالب Keith Schwarz (Standford)

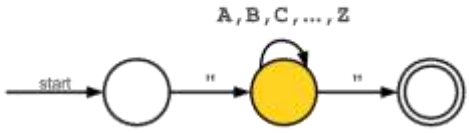
A Simple Automaton

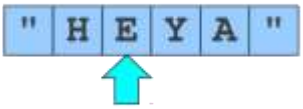




برگرفته از مطالب Keith Schwarz (Standford)

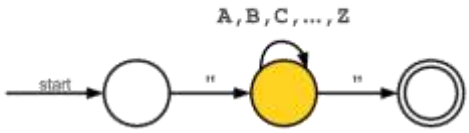
A Simple Automaton

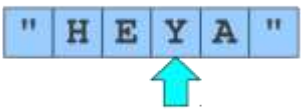




برگرفته از مطالب Keith Schwarz (Standford)

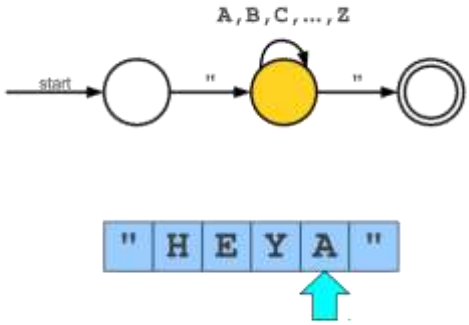
A Simple Automaton





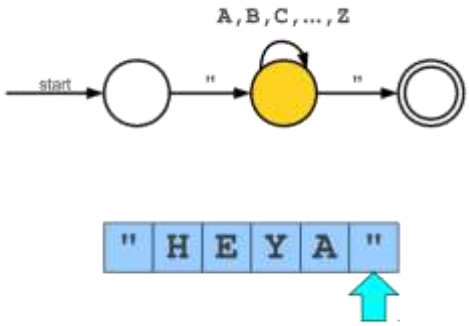
برگرفته از مطالب Keith Schwarz (Standford)

A Simple Automaton



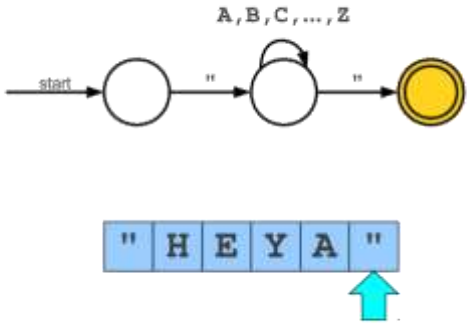
برگرفته از مطالب Keith Schwarz (Standford)

A Simple Automaton



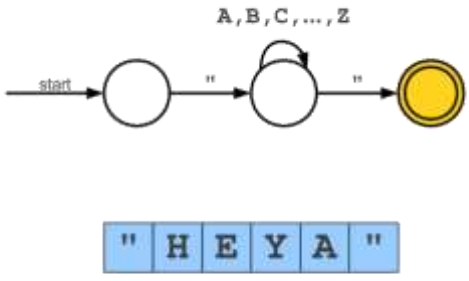
برگرفته از مطالب Keith Schwarz (Standford)

A Simple Automaton



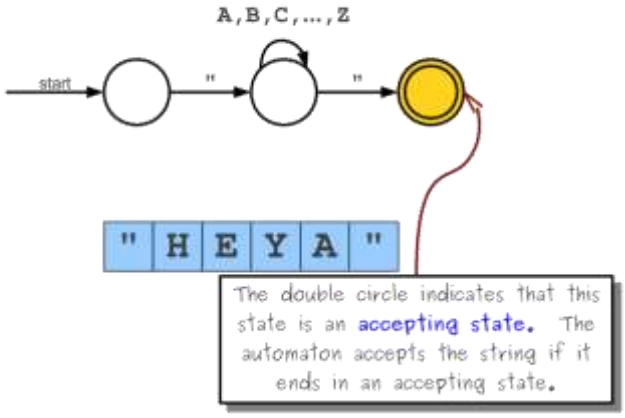
برگرفته از مطالب Keith Schwarz (Standford)

A Simple Automaton



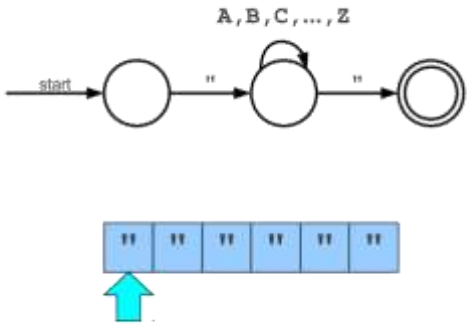
برگرفته از مطالب Keith Schwarz (Standford)

A Simple Automaton



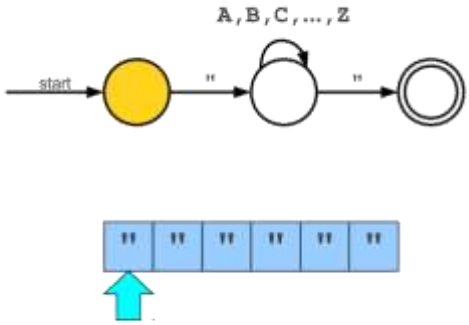
برگرفته از مطالب Keith Schwarz (Standford)

A Simple Automaton



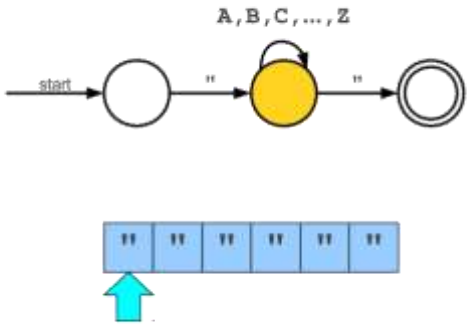
برگرفته از مطالب Keith Schwarz (Standford)

A Simple Automaton



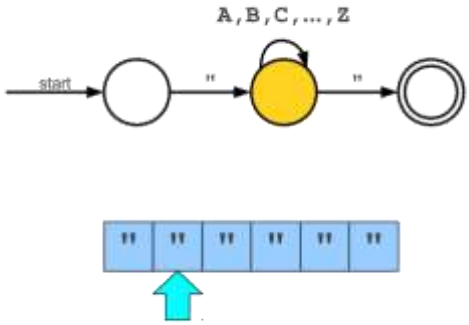
برگرفته از مطالب Keith Schwarz (Standford)

A Simple Automaton



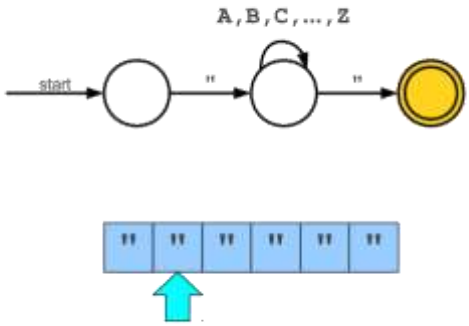
برگرفته از مطالب Keith Schwarz (Standford)

A Simple Automaton



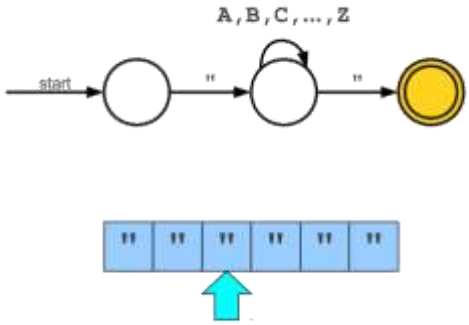
برگرفته از مطالب Keith Schwarz (Standford)

A Simple Automaton



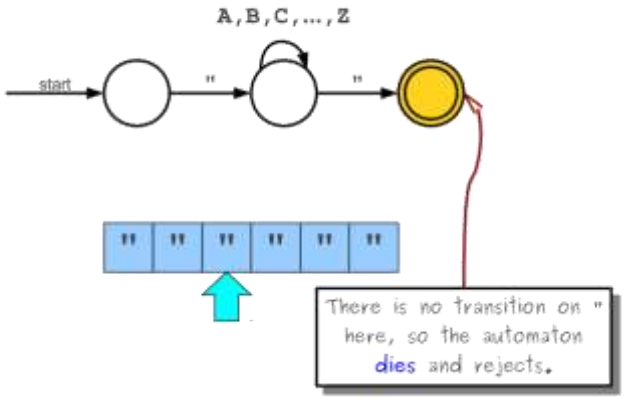
برگرفته از مطالب Keith Schwarz (Standford)

A Simple Automaton



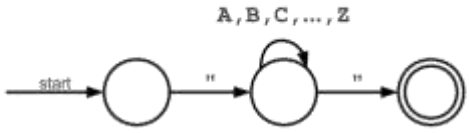
برگرفته از مطالب Keith Schwarz (Standford)

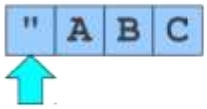
A Simple Automaton



برگرفته از مطالب Keith Schwarz (Standford)

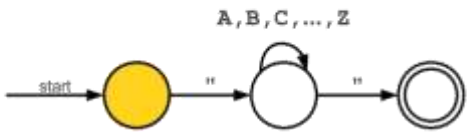
A Simple Automaton

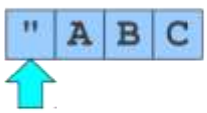




برگرفته از مطالب Keith Schwarz (Standford)

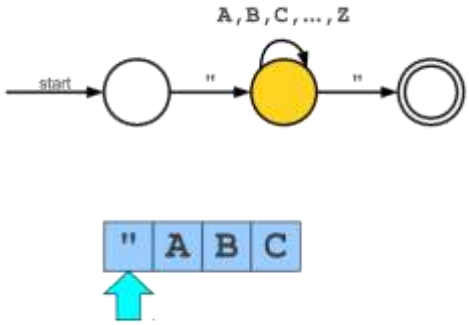
A Simple Automaton





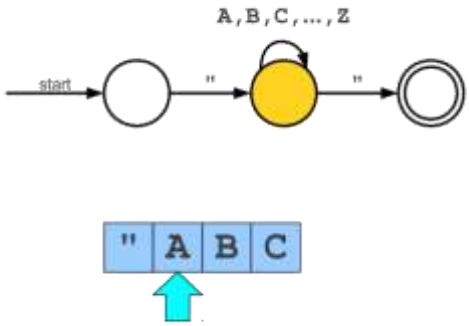
برگرفته از مطالب Keith Schwarz (Standford)

A Simple Automaton



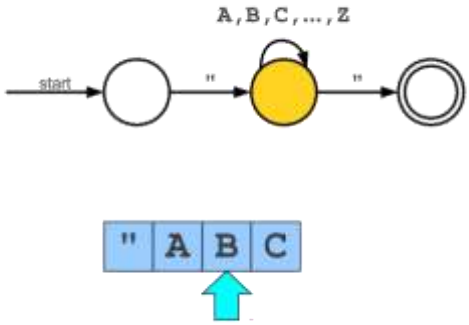
برگرفته از مطالب Keith Schwarz (Standford)

A Simple Automaton



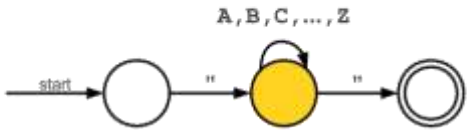
برگرفته از مطالب Keith Schwarz (Standford)

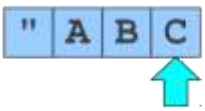
A Simple Automaton



برگرفته از مطالب Keith Schwarz (Standford)

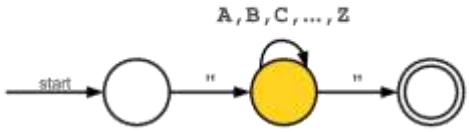
A Simple Automaton





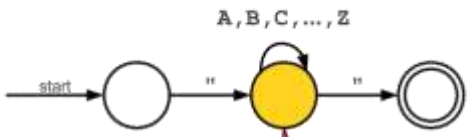
برگرفته از مطالب Keith Schwarz (Standford)

A Simple Automaton



برگرفته از مطالب Keith Schwarz (Standford)

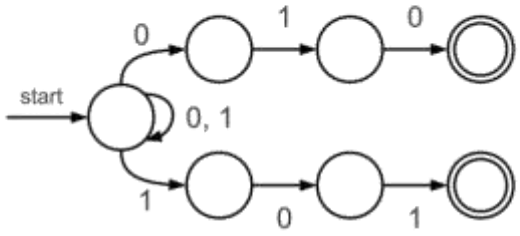
A Simple Automaton



This is not an accepting state, so the automaton rejects.

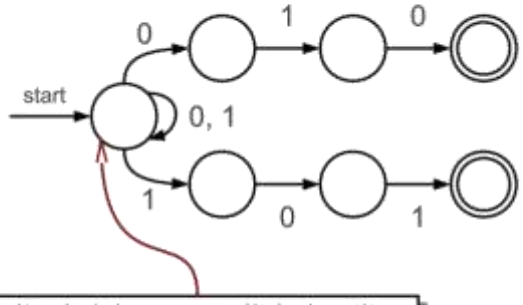
برگرفته از مطالب Keith Schwarz (Standford)

A More Complex Automaton



برگرفته از مطالب Keith Schwarz (Standford)

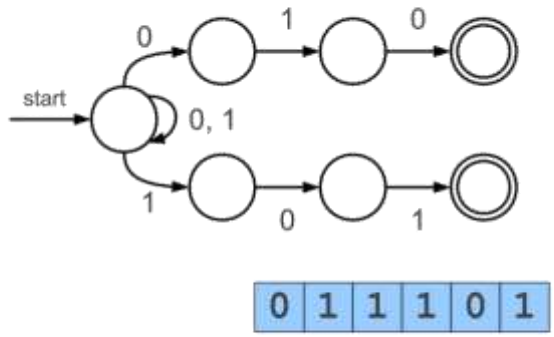
A More Complex Automaton



Notice that there are multiple transitions defined here on 0 and 1. If we read a 0 or 1 here, we follow *both* transitions and enter multiple states.

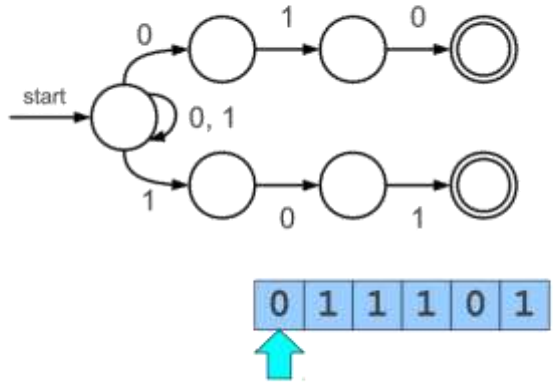
برگرفته از مطالب Keith Schwarz (Standford)

A More Complex Automaton



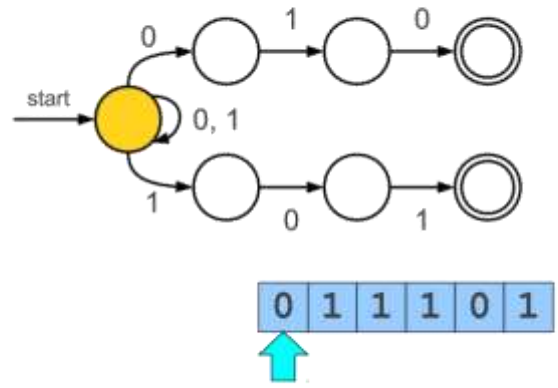
برگرفته از مطالب Keith Schwarz (Standford)

A More Complex Automaton



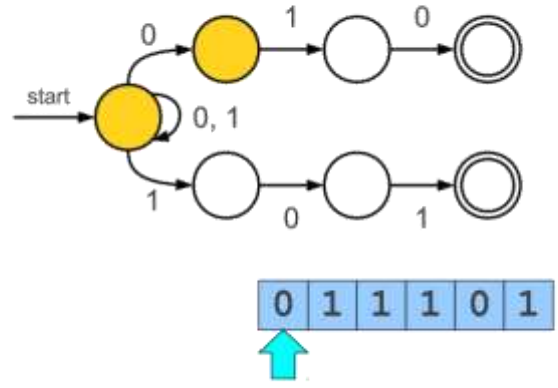
برگرفته از مطالب Keith Schwarz (Standford)

A More Complex Automaton



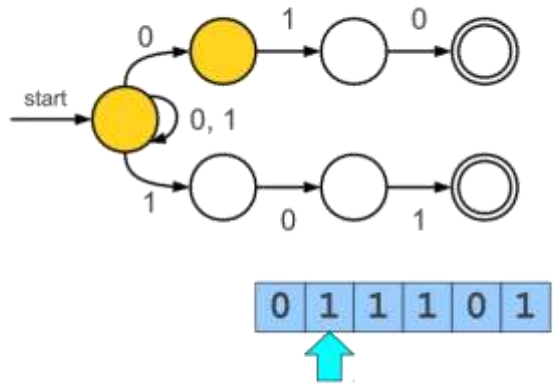
برگرفته از مطالب Keith Schwarz (Standford)

A More Complex Automaton



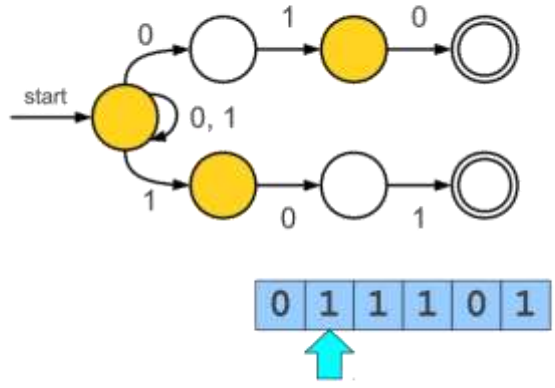
برگرفته از مطالب Keith Schwarz (Standford)

A More Complex Automaton



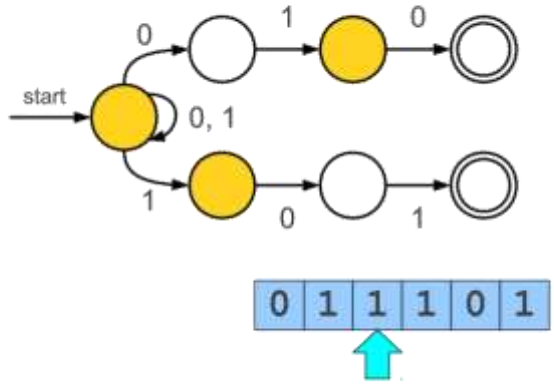
برگرفته از مطالب Keith Schwarz (Standford)

A More Complex Automaton



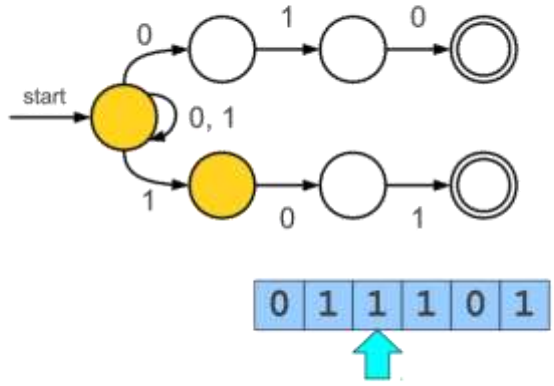
برگرفته از مطالب Keith Schwarz (Standford)

A More Complex Automaton



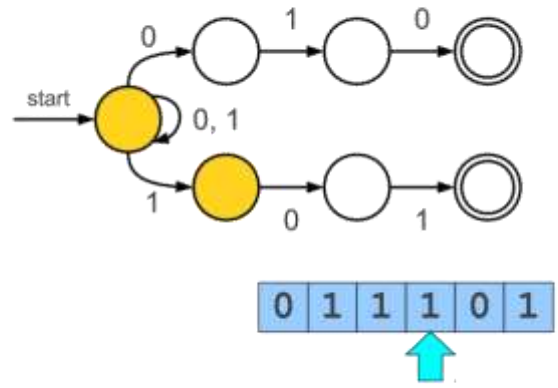
برگرفته از مطالب Keith Schwarz (Standford)

A More Complex Automaton



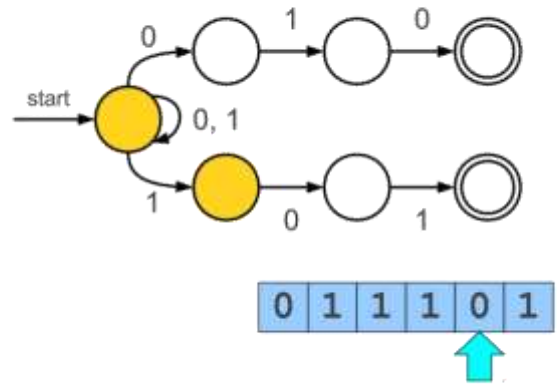
برگرفته از مطالب Keith Schwarz (Standford)

A More Complex Automaton



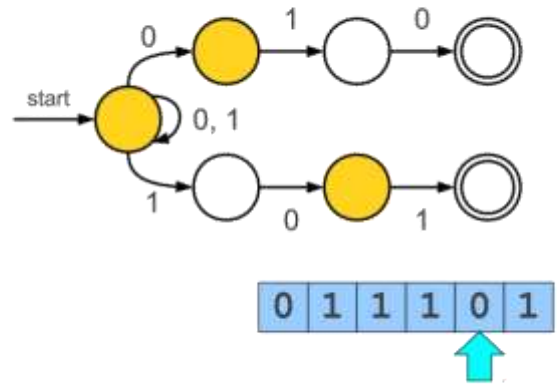
برگرفته از مطالب Keith Schwarz (Standford)

A More Complex Automaton



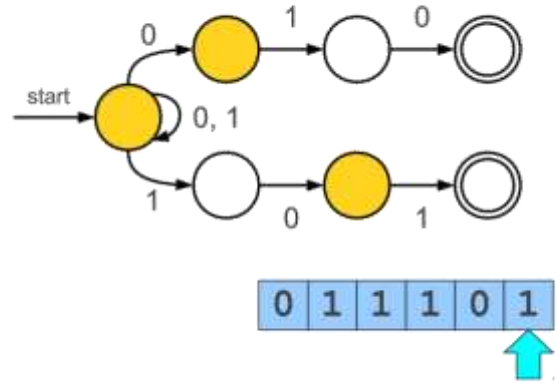
برگرفته از مطالب Keith Schwarz (Standford)

A More Complex Automaton



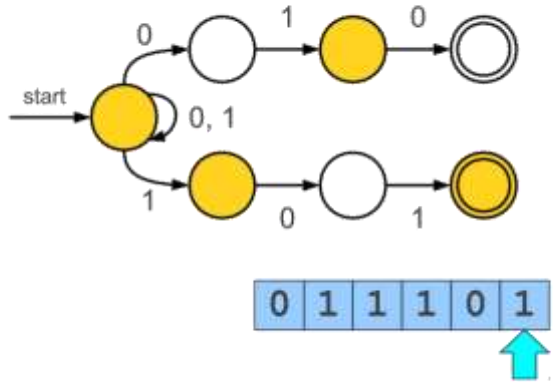
برگرفته از مطالب Keith Schwarz (Standford)

A More Complex Automaton



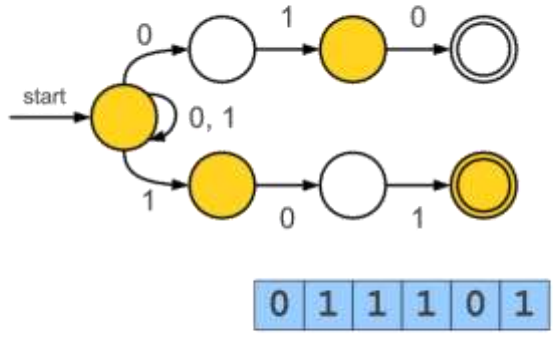
برگرفته از مطالب Keith Schwarz (Standford)

A More Complex Automaton



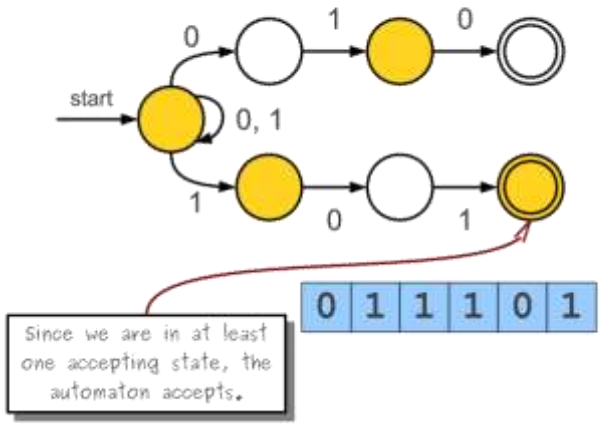
برگرفته از مطالب Keith Schwarz (Standford)

A More Complex Automaton



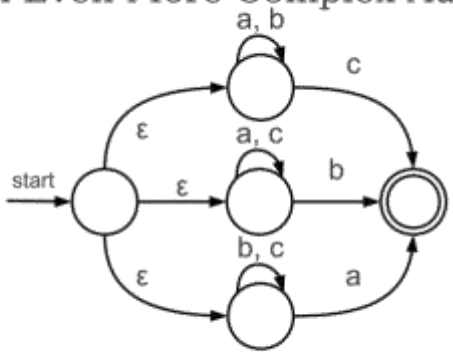
برگرفته از مطالب Keith Schwarz (Standford)

A More Complex Automaton



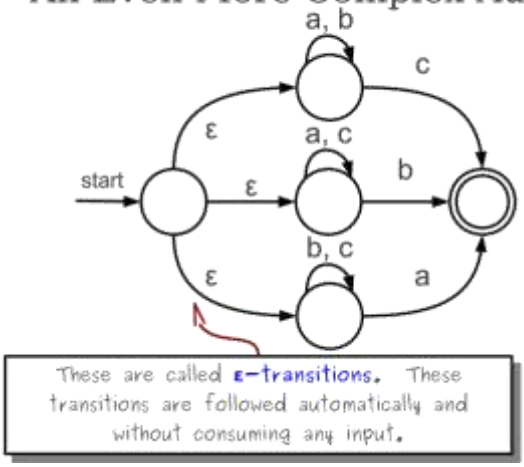
برگرفته از مطالب Keith Schwarz (Standford)

An Even More Complex Automaton



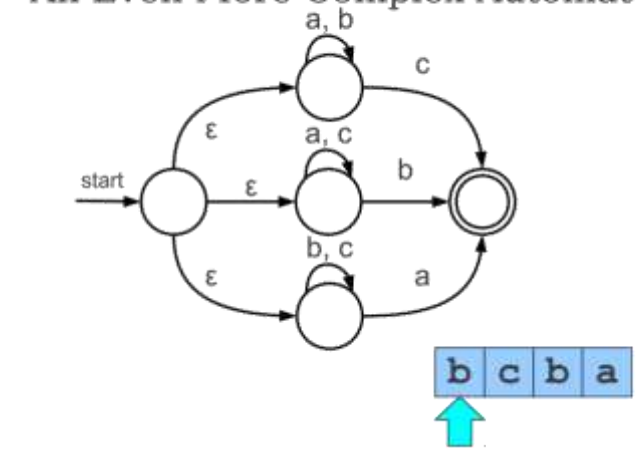
برگرفته از مطالب Keith Schwarz (Standford)

An Even More Complex Automaton



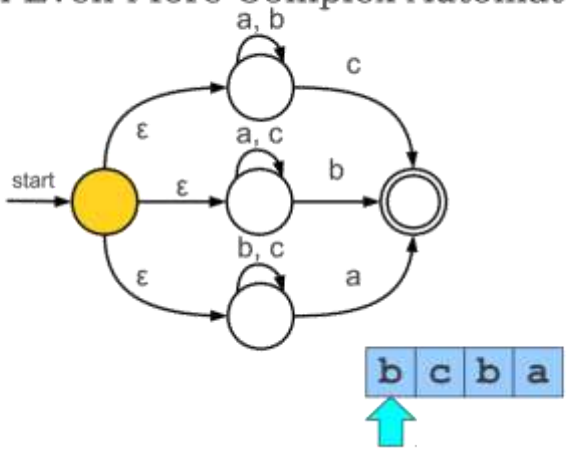
برگرفته از مطالب Keith Schwarz (Standford)

An Even More Complex Automaton



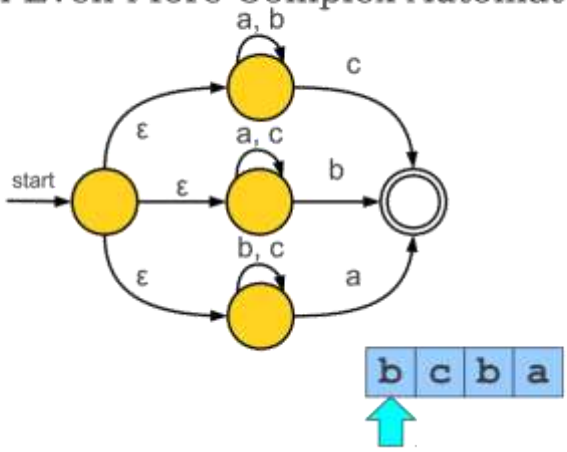
برگرفته از مطالب Keith Schwarz (Standford)

An Even More Complex Automaton



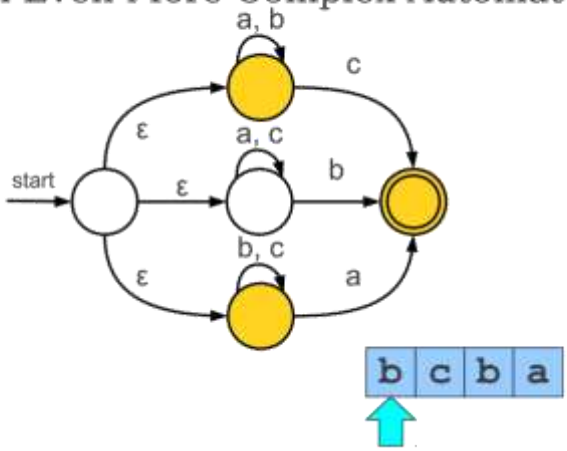
برگرفته از مطالب Keith Schwarz (Standford)

An Even More Complex Automaton



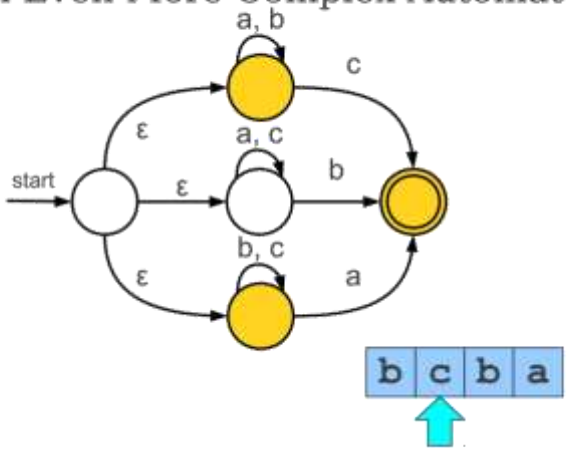
برگرفته از مطالب Keith Schwarz (Standford)

An Even More Complex Automaton



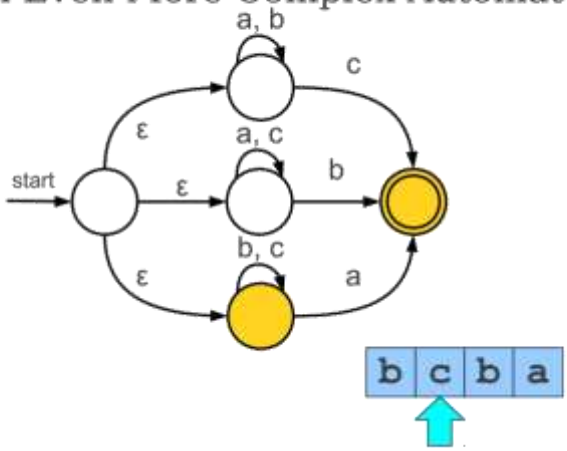
برگرفته از مطالب Keith Schwarz (Standford)

An Even More Complex Automaton



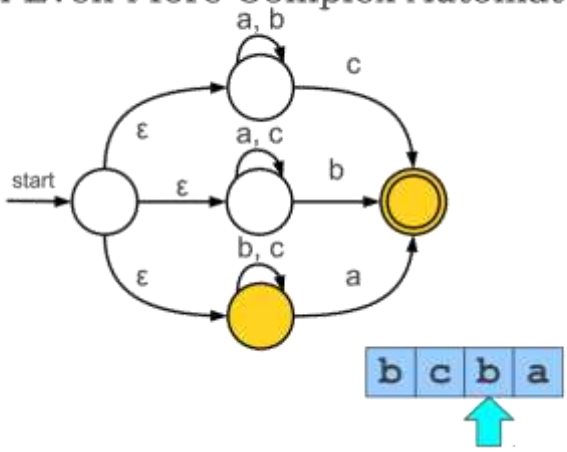
برگرفته از مطالب Keith Schwarz (Standford)

An Even More Complex Automaton



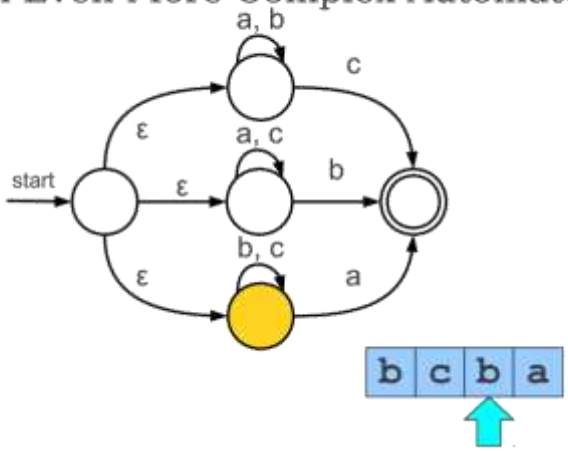
برگرفته از مطالب Keith Schwarz (Standford)

An Even More Complex Automaton



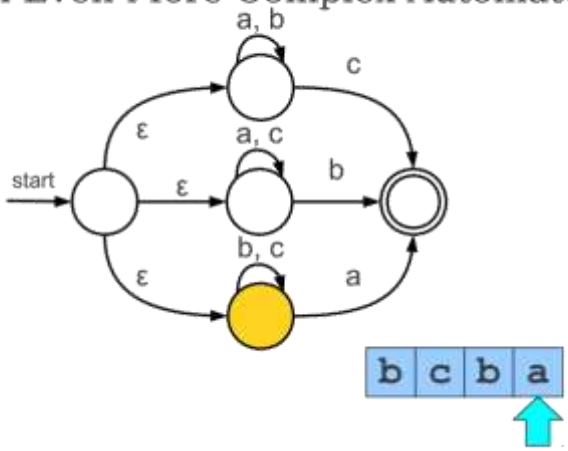
برگرفته از مطالب Keith Schwarz (Standford)

An Even More Complex Automaton



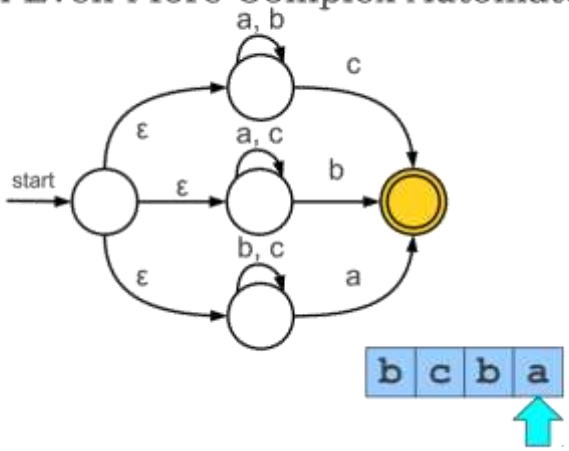
برگرفته از مطالب Keith Schwarz (Standford)

An Even More Complex Automaton



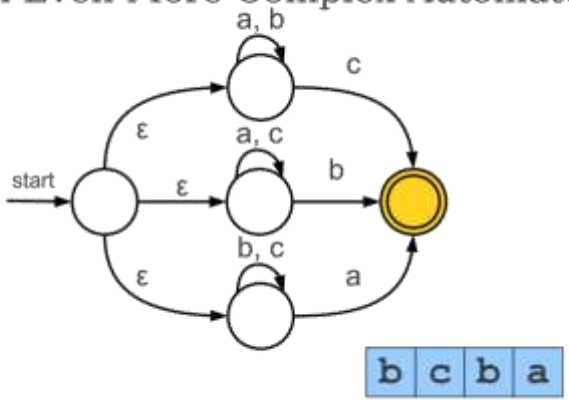
برگرفته از مطالب Keith Schwarz (Standford)

An Even More Complex Automaton



برگرفته از مطالب Keith Schwarz (Standford)

An Even More Complex Automaton



برگرفته از مطالب Keith Schwarz (Standford)

تبدیل عبارت منظم به ماشین حالت

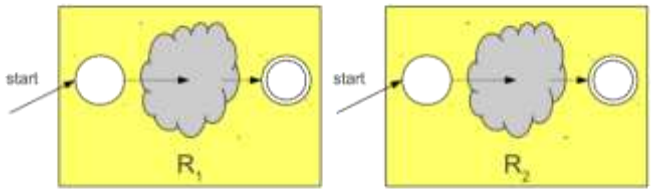
Base Cases

Automaton for ε

Automaton for single character a

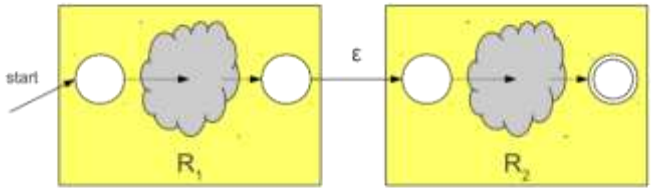
برگرفته از مطالب Keith Schwarz (Standford)

Construction for R_1R_2



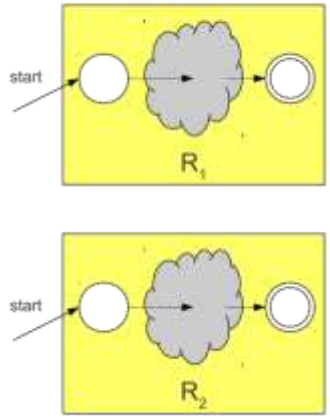
برگرفته از مطالب Keith Schwarz (Standford)

Construction for R_1R_2



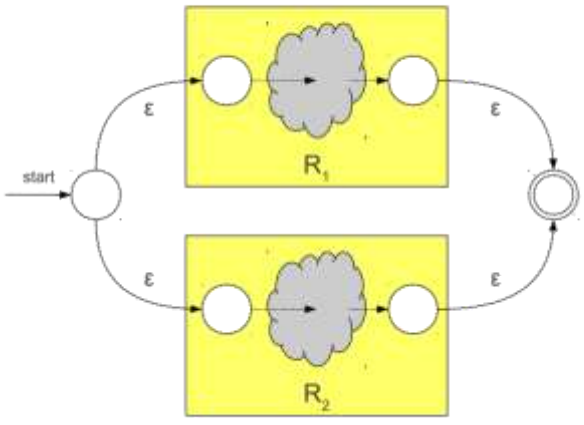
برگرفته از مطالب Keith Schwarz (Standford)

Construction for $R_1 \mid R_2$



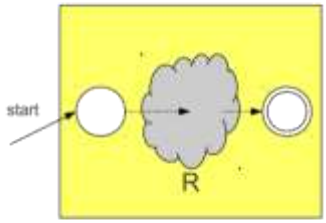
برگرفته از مطالب Keith Schwarz (Standford)

Construction for $R_1 \mid R_2$



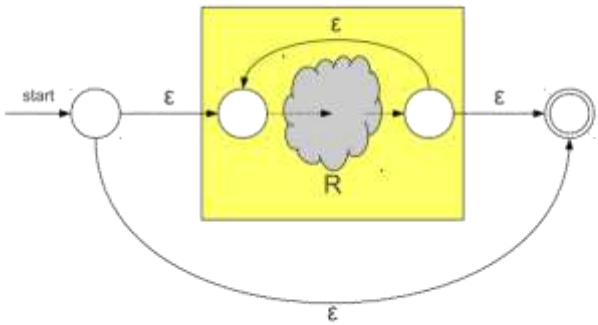
برگرفته از مطالب Keith Schwarz (Standford)

Construction for R*



برگرفته از مطالب Keith Schwarz (Standford)

Construction for R*



برگرفته از مطالب Keith Schwarz (Standford)

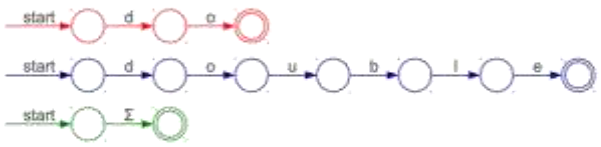
سوال؟

- ماشین حالت معادل عبارت زیر؟
- $0?(1^+0?)^*$
- عبارت فوق چه زبانی را توصیف می‌کند؟
  - رشته‌هایی که دو صفر پشت سر هم نداشته باشند.

پیاده‌سازی پویشگر

- ایده: پیاده‌سازی پویشگر با استفاده از ماشین حالت.
  - تمامی الگوها تبدیل به ماشین حالت شوند.
  - رشته ورودی به ترتیب در همه ماشین‌ها بررسی شود.
  - ماشینی که رشته را پذیرفت، توکن را مشخص می‌کند.

```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```



برگرفته از مطالب Keith Schwarz (Standford)

مشکلات تشخیص توکن‌ها

- یک عبارت منظم بخشی از یک لغت را تشخیص دهد.

برگرفته از مطالب Keith Schwarz (Standford)

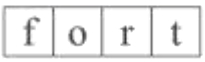
Lexing Ambiguities

```
T_For           for
T_Identifier    [A-Za-z_][A-Za-z0-9_]*
```

برگرفته از مطالب Keith Schwarz (Standford)

Lexing Ambiguities

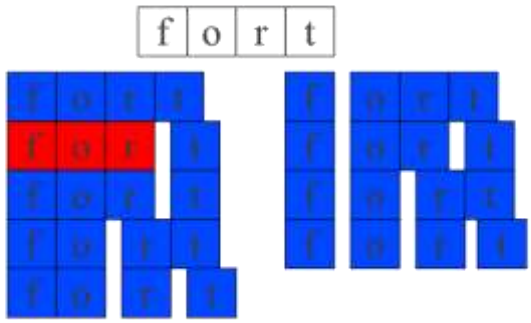
```
T_For           for
T_Identifier    [A-Za-z_][A-Za-z0-9_]*
```



برگرفته از مطالب Keith Schwarz (Standford)

Lexing Ambiguities

```
T_For          for
T_Identifier   [A-Za-z_][A-Za-z0-9_]*
```



برگرفته از مطالب Keith Schwarz (Standford)

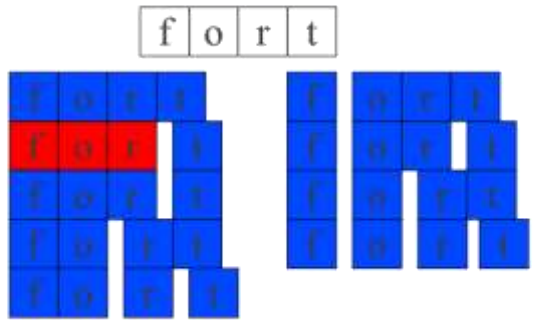
مشکلات تشخیص توکن‌ها

- یک عبارت منظم بخشی از یک لغت را تشخیص دهد.
  - راه حل: استراتژی حداکثر طول
  - لغت مربوط به یک توکن آنست که بیشترین طول ممکن را دارد.

برگرفته از مطالب Keith Schwarz (Standford)

Lexing Ambiguities

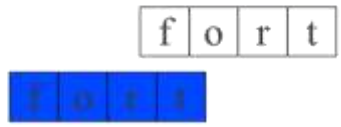
```
T_For          for
T_Identifier   [A-Za-z_][A-Za-z0-9_]*
```



برگرفته از مطالب Keith Schwarz (Standford)

Lexing Ambiguities

```
T_For          for
T_Identifier   [A-Za-z_][A-Za-z0-9_]*
```



برگرفته از مطالب Keith Schwarz (Standford)

پیاده‌سازی روش حداکثر طول

- همه عبارات منظم را به NFA تبدیل می کنیم.
- همزمان تمامی NFAهای حاصل را اجرا می کنیم.
  - در صورت رسیدن به حالت پذیرش متوقف نمی‌شویم.
    - آخرین لغت پذیرفته شده را ذخیره می‌کنیم.
  - وقتی به شکست برخوردیم به وضعیت آخرین حالت پذیرش برمی‌گردیم.

برگرفته از مطالب Keith Schwarz (Standford)

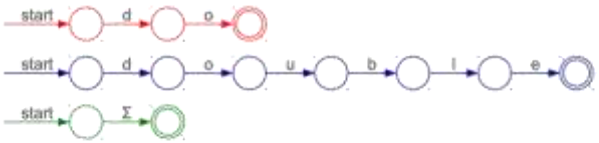
Implementing Maximal Munch

```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```

برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

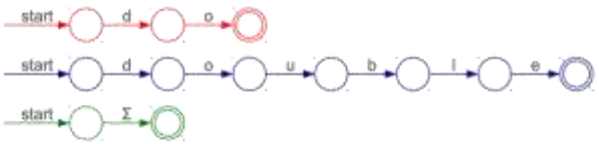
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```



برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```

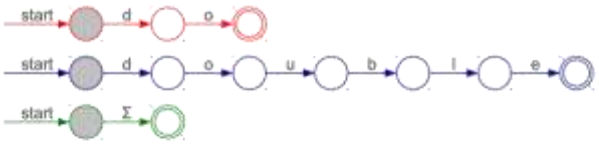


D	O	U	B	D	O	U	B	L	E

برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```

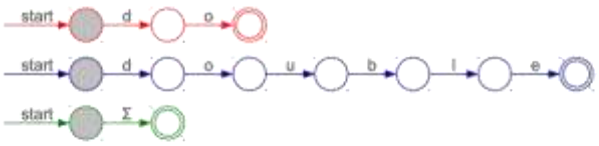


D	O	U	B	D	O	U	B	L	E

برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```

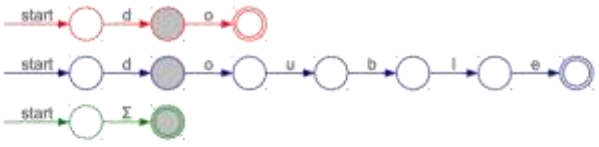


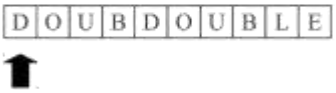
D	O	U	B	D	O	U	B	L	E

برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```

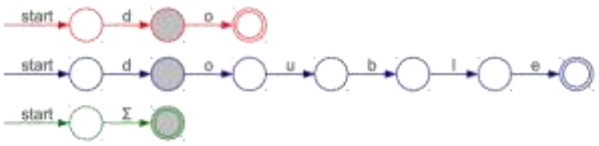


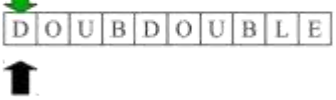


برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```

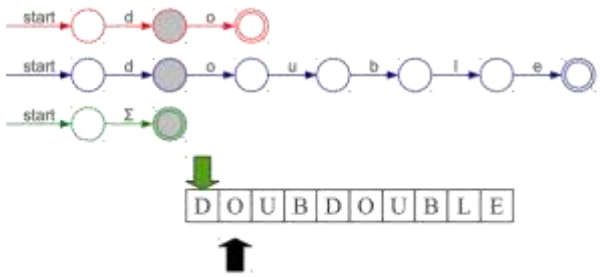




برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

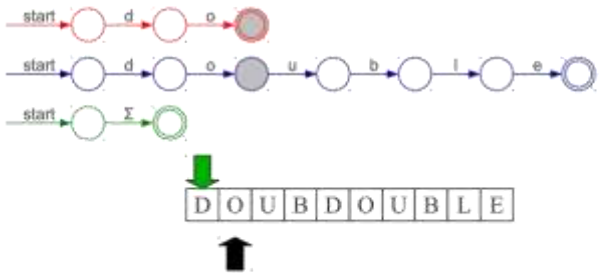
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```



برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

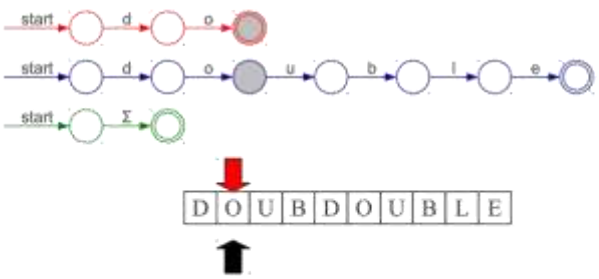
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```



برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

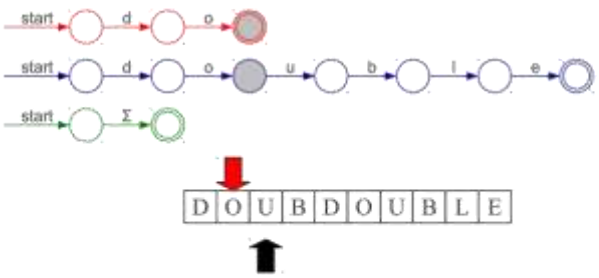
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```



برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

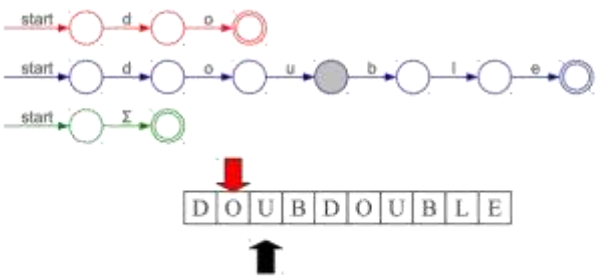
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```

برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

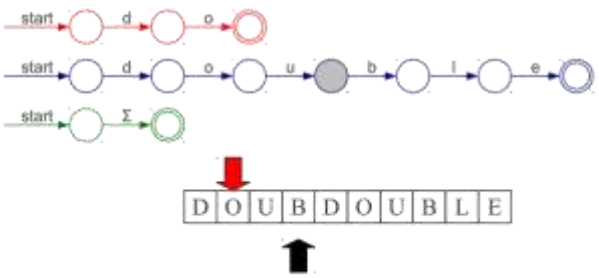
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```



برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

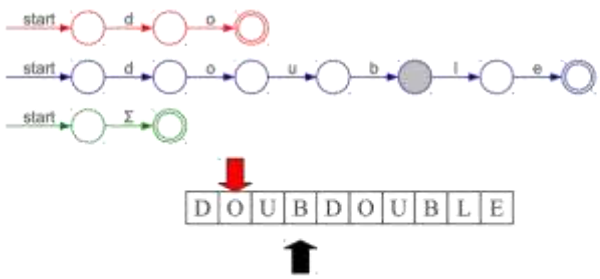
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```

برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

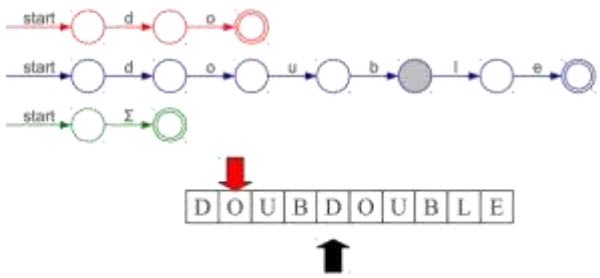
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```



برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

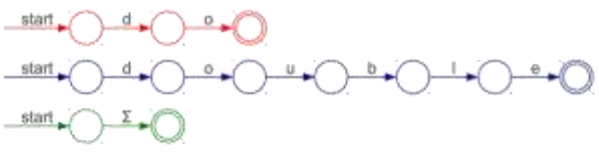
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```

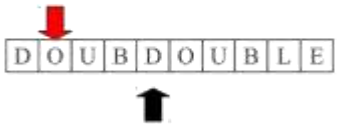


برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```

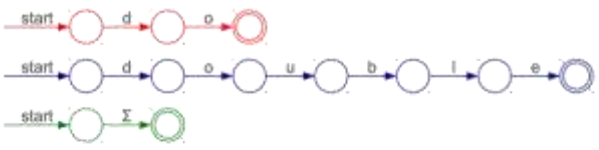


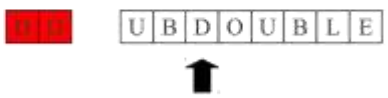


برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```

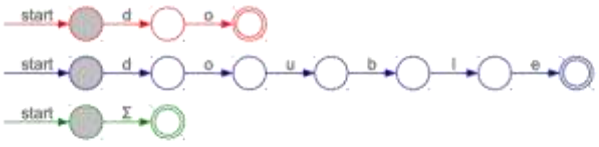


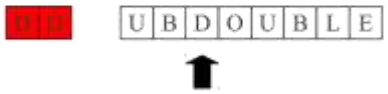


برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```

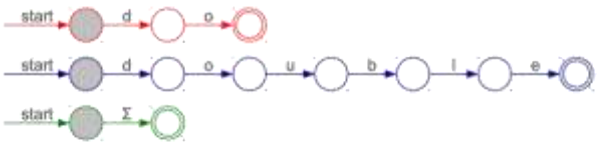


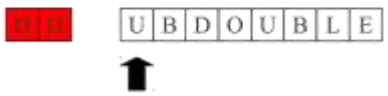


برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```

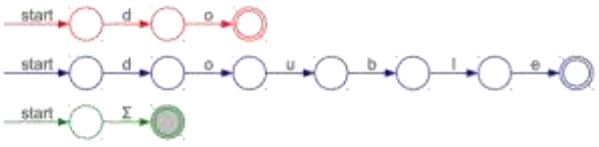



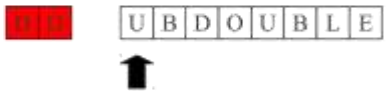


برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```

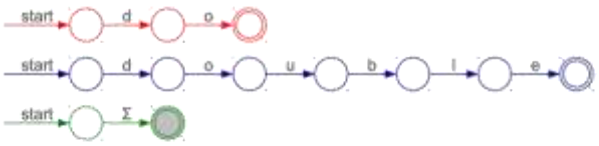


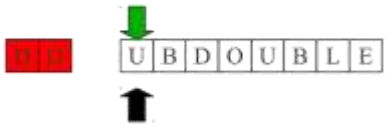


برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```

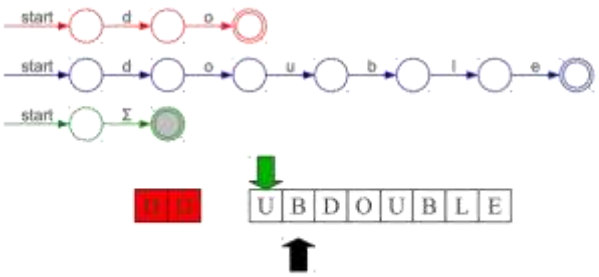




برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

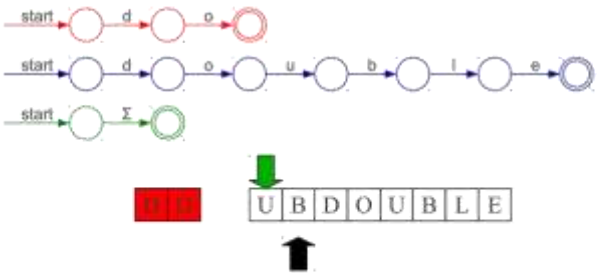
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```



برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

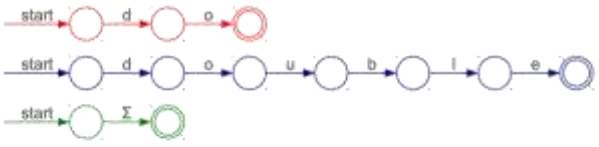
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```

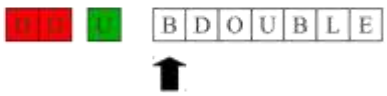



برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```

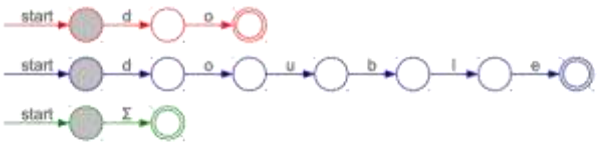


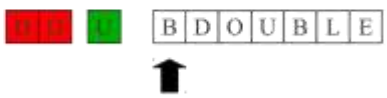


برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```

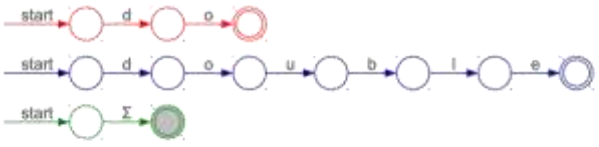


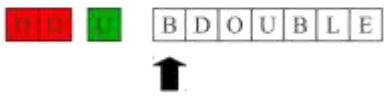


برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```

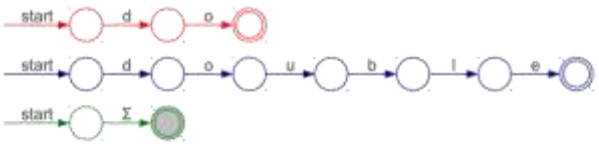


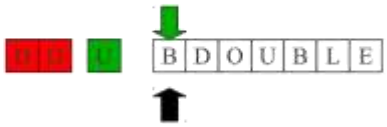


برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```

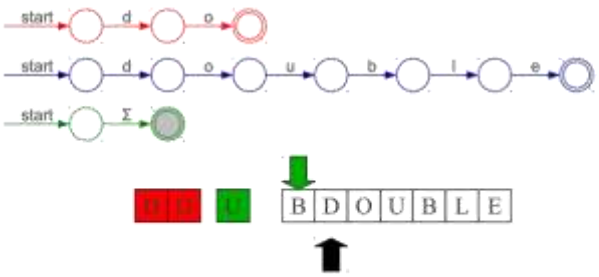




برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

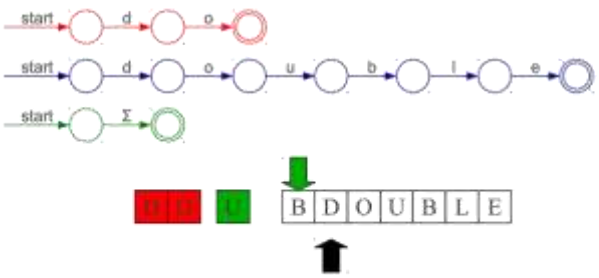
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```



برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

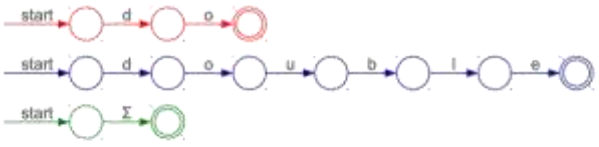
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```

برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

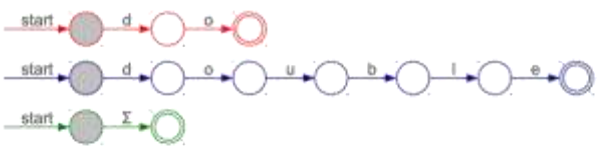
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```



برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

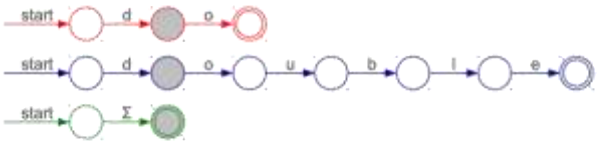
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```

برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

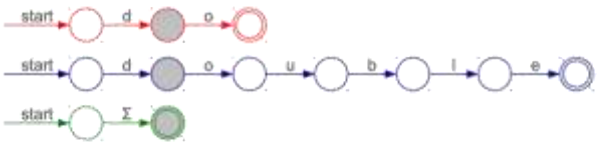
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```

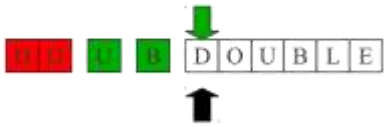


برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```

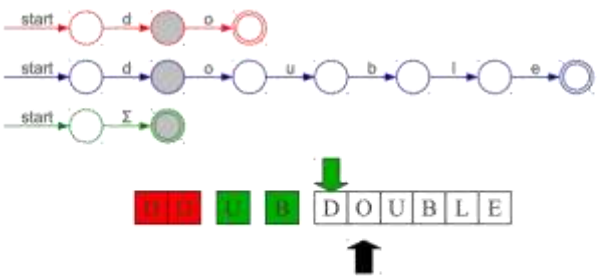




برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

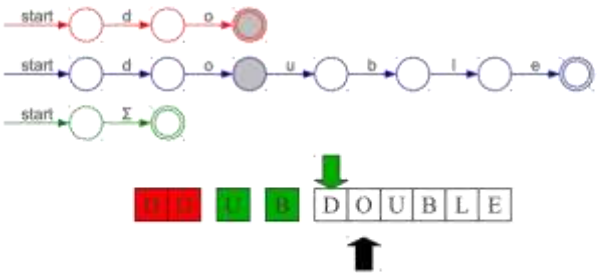
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```



برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

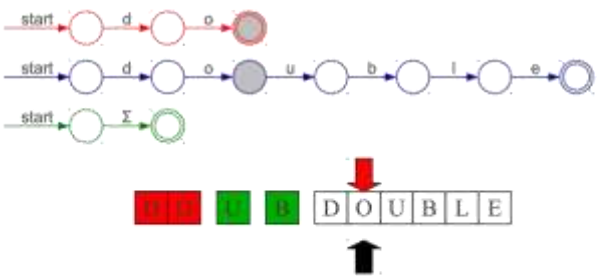
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```

برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

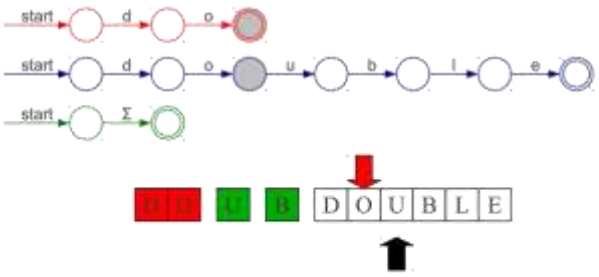
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```



برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

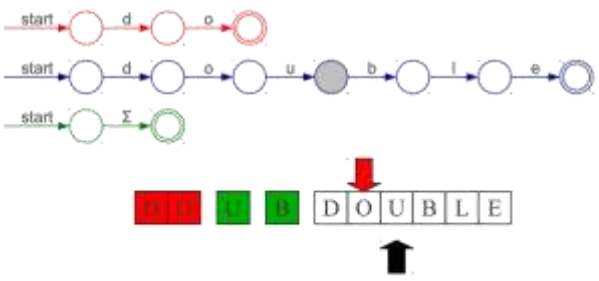
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```

برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

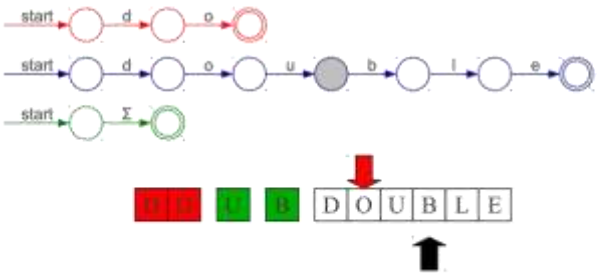
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```



برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

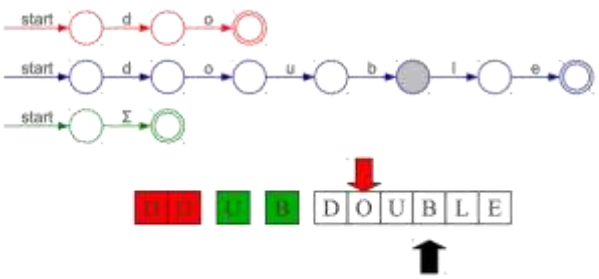
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```

برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

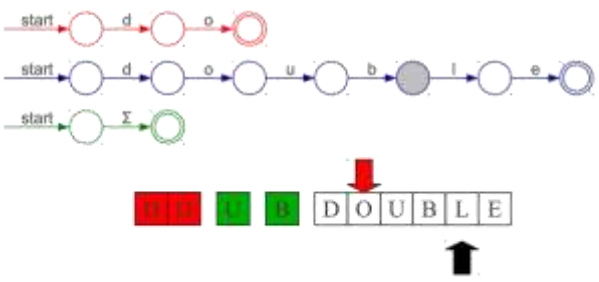
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```



برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

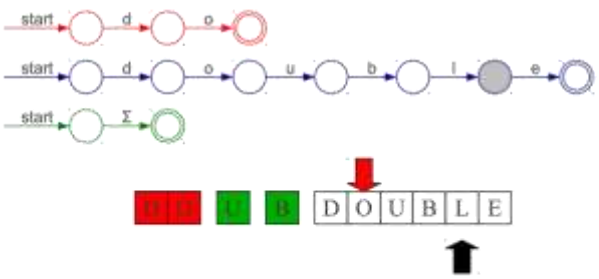
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```

برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

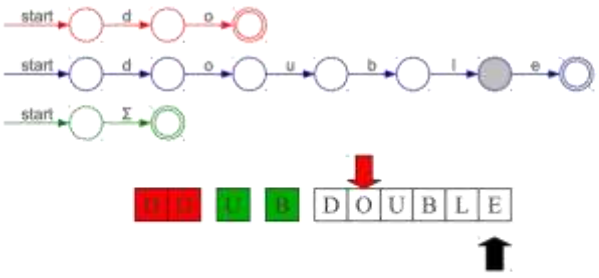
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```



برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

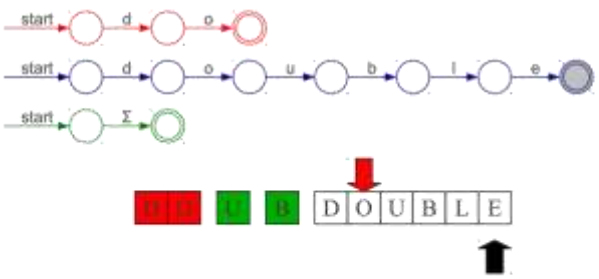
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```

برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

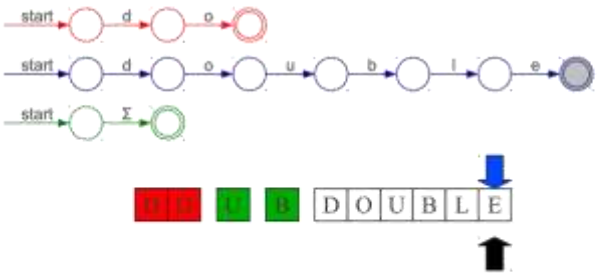
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```



برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

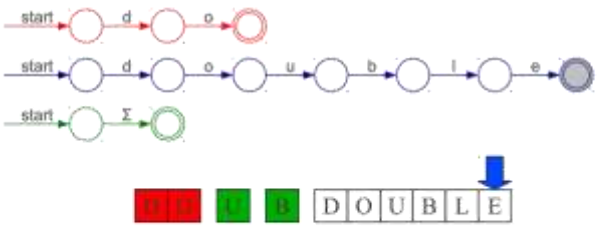
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```



برگرفته از مطالب Keith Schwarz (Standford)

Implementing Maximal Munch

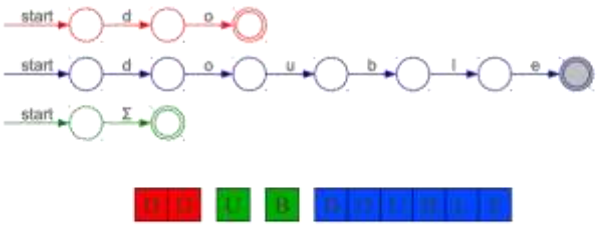
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```



برگرفته از مطالب Keith Schwarz (Standford)

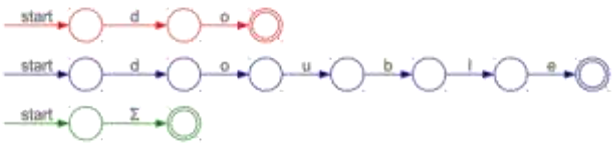
Implementing Maximal Munch

```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```



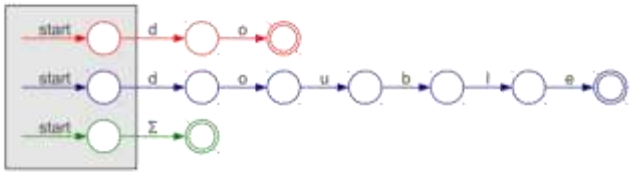
برگرفته از مطالب Keith Schwarz (Standford)

A Minor Simplification



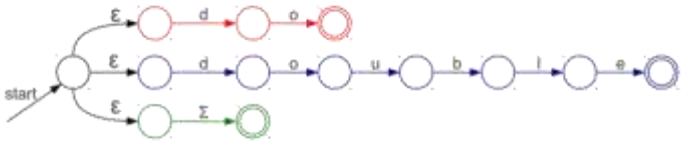
برگرفته از مطالب Keith Schwarz (Standford)

A Minor Simplification

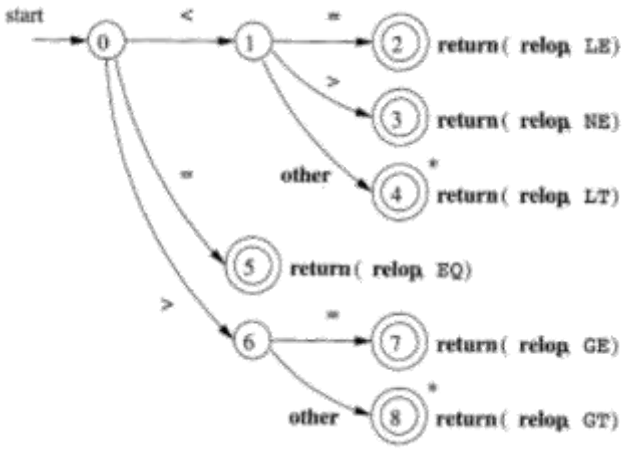


برگرفته از مطالب Keith Schwarz (Standford)

A Minor Simplification



برگرفته از مطالب Keith Schwarz (Standford)



مشکلات تشخیص توکن‌ها

- یک عبارت منظم بخشی از یک لغت را تشخیص دهد.
  - راه حل: استراتژی حداکثر طول
  - لغت مربوط به یک توکن آنست که بیشترین طول ممکن را دارد.
- یک لغت توسط دو یا چند عبارت منظم تشخیص داده شود.

برگرفته از مطالب Keith Schwarz (Standford)

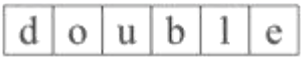
Other Conflicts

```
T_Do          do
T_Double      double
T_Identifier  [A-Za-z_][A-Za-z0-9_]*
```

برگرفته از مطالب Keith Schwarz (Standford)

Other Conflicts

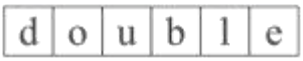
```
T_Do          do
T_Double      double
T_Identifier  [A-Za-z_][A-Za-z0-9_]*
```



برگرفته از مطالب Keith Schwarz (Standford)

Other Conflicts

```
T_Do          do
T_Double      double
T_Identifier  [A-Za-z_][A-Za-z0-9_]*
```



برگرفته از مطالب Keith Schwarz (Standford)

مشکلات تشخیص توکن‌ها

- یک عبارت منظم بخشی از یک لغت را تشخیص دهد.
  - راه حل: استراتژی حداکثر طول
- یک لغت توسط دو یا چند عبارت منظم تشخیص داده شود.
  - راه حل: عبارت منظمی که در لیست جلوتر قرار دارد، در نظر گرفته می‌شود.

برگرفته از مطالب Keith Schwarz (Standford)

مشکلات تشخیص توکن‌ها

- یک عبارت منظم بخشی از یک لغت را تشخیص دهد.
  - راه حل: استراتژی حداکثر طول
- یک لغت توسط دو یا چند عبارت منظم تشخیص داده شود.
  - راه حل: عبارت منظمی که در لیست جلوتر قرار دارد، در نظر گرفته می‌شود.
- تشخیص توکن در زبانهایی مانند FORTRAN و PL/1
  - جنس توکن به لغت‌های آینده بستگی دارد.

برگرفته از مطالب Keith Schwarz (Standford)

مشکلات تشخیص توکن‌ها

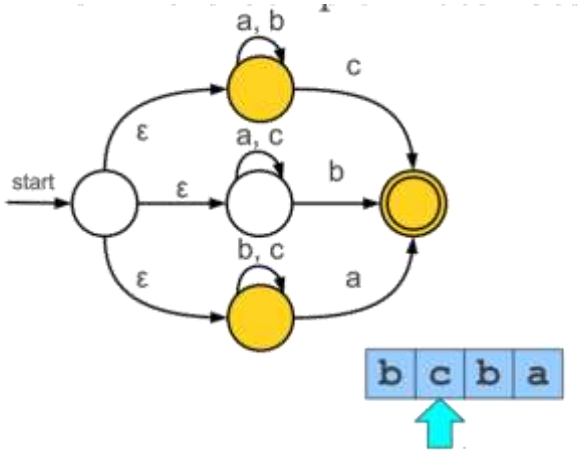
- یک عبارت منظم بخشی از یک لغت را تشخیص دهد.
  - راه حل: استراتژی حداکثر طول
- یک لغت توسط دو یا چند عبارت منظم تشخیص داده شود.
  - راه حل: عبارت منظمی که در لیست جلوتر قرار دارد، در نظر گرفته می‌شود.
- تشخیص توکن در زبانهایی مانند FORTRAN و PL/1
  - راه حل: نگاه به جلو، خواندن کاراکترهای اضافی و بازگشت.
  - در ابزارهای پویش یه نام Lookahead operator شناخته می‌شود.

برگرفته از مطالب Keith Schwarz (Standford)

مشکلات تشخیص توکن‌ها

- یک عبارت منظم بخشی از یک لغت را تشخیص دهد.
  - راه حل: استراتژی حداکثر طول
  - لغت مربوط به یک توکن آنست که بیشترین طول ممکن را دارد.
- یک لغت توسط دو یا چند عبارت منظم تشخیص داده شود.
  - راه حل: عبارت منظمی که در لیست جلوتر قرار دارد، در نظر گرفته می‌شود.
- تشخیص توکن در زبانهایی مانند FORTRAN و PL/1
  - راه حل: نگاه به جلو، خواندن کاراکترهای اضافی و بازگشت.
- مرتبه اجرای ماشین حالت؟

برگرفته از مطالب Keith Schwarz (Standford)



برگرفته از مطالب Keith Schwarz (Standford)

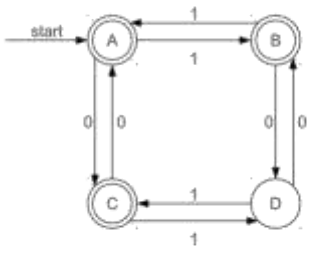
مشکلات تشخیص توکن‌ها

- یک عبارت منظم بخشی از یک لغت را تشخیص دهد.
  - راه حل: استراتژی حداکثر طول
  - لغت مربوط به یک توکن آنست که بیشترین طول ممکن را دارد.
- یک لغت توسط دو یا چند عبارت منظم تشخیص داده شود.
  - راه حل: عبارت منظمی که در لیست جلوتر قرار دارد، در نظر گرفته می‌شود.
- تشخیص توکن در زبانهایی مانند FORTRAN و PL/1
  - راه حل: نگاه به جلو، خواندن کاراکترهای اضافی و بازگشت.
- مرتبه اجرای ماشین حالت؟
- O(mn^2)
  - n: number of state
  - m: length of lexeme

مشکلات تشخیص توکن‌ها

- یک عبارت منظم بخشی از یک لغت را تشخیص دهد.
  - راه حل: استراتژی حداکثر طول
  - لغت مربوط به یک توکن آنست که بیشترین طول ممکن را دارد.
- یک لغت توسط دو یا چند عبارت منظم تشخیص داده شود.
  - راه حل: عبارت منظمی که در لیست جلوتر قرار دارد، در نظر گرفته می‌شود.
- تشخیص توکن در زبانهایی مانند FORTRAN و PL/1
  - راه حل: نگاه به جلو، خواندن کاراکترهای اضافی و بازگشت.
- مرتبه اجرای ماشین حالت، مناسب اجرای عملیاتی نیست.
  - ایده: اگر هر لحظه فقط در یک حالت باشیم.
  - ماشین حالت قطعی (مرتبه اجرا: O(m))

A Sample DFA



	0	1
A	C	B
B	D	A
C	A	D
D	B	C

برگرفته از مطالب Keith Schwarz (Standford)

تبدیل NFA به DFA

- NFA در هر لحظه ممکن است در چندین حالت باشد.
- DFA در هر لحظه فقط در یک حالت است.
- ایده: روشی بیابیم که DFA، NFA را شبیه‌سازی کند.

برگرفته از مطالب Keith Schwarz (Standford)

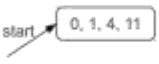
From NFA to DFA

برگرفته از مطالب Keith Schwarz (Standford)

From NFA to DFA

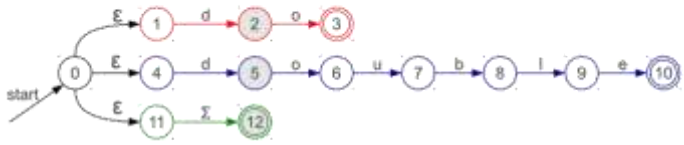
برگرفته از مطالب Keith Schwarz (Standford)

From NFA to DFA



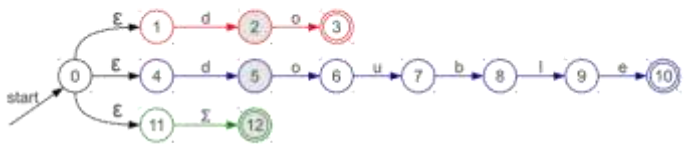
برگرفته از مطالب Keith Schwarz (Standford)

From NFA to DFA



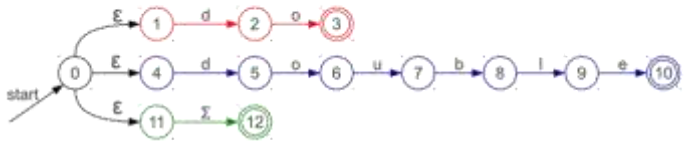
برگرفته از مطالب Keith Schwarz (Standford)

From NFA to DFA



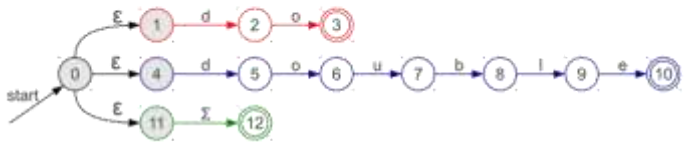
برگرفته از مطالب Keith Schwarz (Standford)

From NFA to DFA



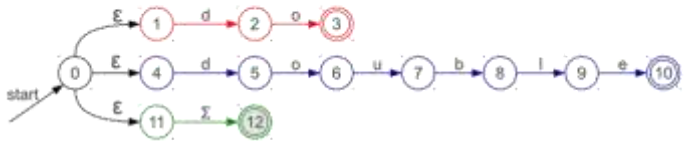
برگرفته از مطالب Keith Schwarz (Standford)

From NFA to DFA



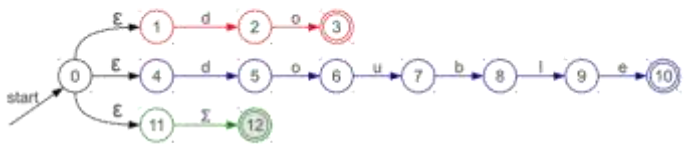
برگرفته از مطالب Keith Schwarz (Standford)

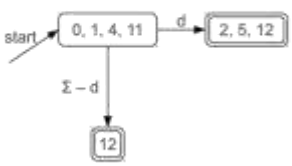
From NFA to DFA



برگرفته از مطالب Keith Schwarz (Standford)

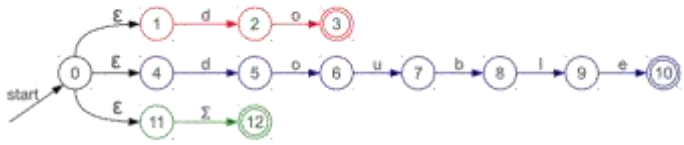
From NFA to DFA

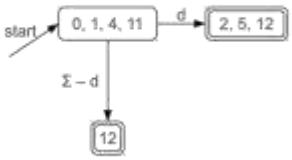




برگرفته از مطالب Keith Schwarz (Standford)

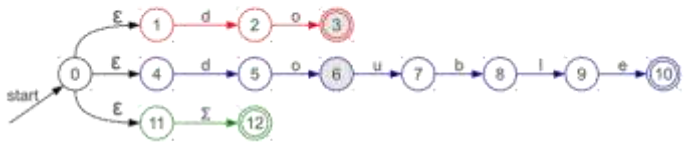
From NFA to DFA

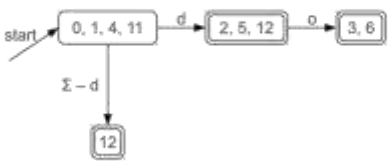




برگرفته از مطالب Keith Schwarz (Standford)

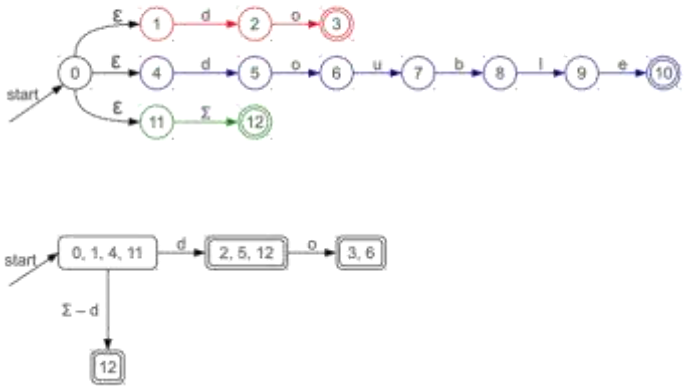
From NFA to DFA





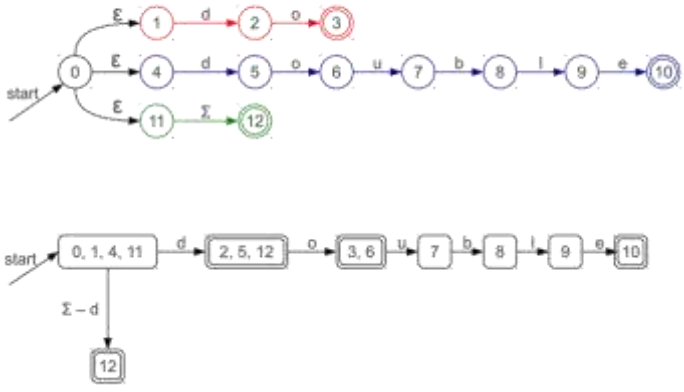
برگرفته از مطالب Keith Schwarz (Standford)

From NFA to DFA



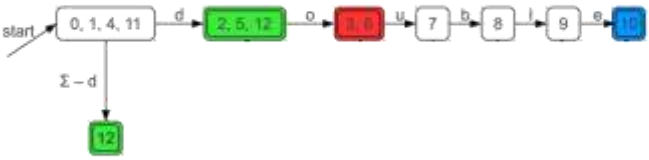
برگرفته از مطالب Keith Schwarz (Standford)

From NFA to DFA



برگرفته از مطالب Keith Schwarz (Standford)

From NFA to DFA



برگرفته از مطالب Keith Schwarz (Standford)

مسئله

- رشته ورودی:
- dictatorial
- جزییات تحلیل‌گر لغوی:
- dict (1)
  dictator (2)
  [a-z]* (3)
  dictatorial (4)
- توکن تشخیص داده شده برای رشته ورودی؟

برگرفته از مطالب Keith Schwarz (Standford)

Q?

- Consider the string ***abbbaacc***. Which of the following lexical specifications produces the tokenization ***ab /bb /a /acc*** ?

❑ **b+**
ab*
ac*

❑ **ab**
b+
ac*

Q?

- Consider the string ***abbbaacc***. Which of the following lexical specifications produces the tokenization ***ab /bb /a /acc*** ?

❑ **b+** tokenized as ***abbb /a /acc*** .
ab*
ac*

❑ **ab** ☑
b+
ac*

Q?

- Given the following lexical specification:

  a(ba)*
  b*(ab)*
  abd
  d+
  which of the following statements is true?

- ***ababddababa*** will be tokenized as:
  ab /abd /d /ababa
- ***ababdddd*** will be tokenized as:
  abab /dddd
- ***dddabbabab*** will be tokenized as:
  ddd /a /bbabab
- ***babad*** will be tokenized as:
  bab /a /d

Q?

- Given the following lexical specification:

  a(ba)*
  b*(ab)*
  abd
  d+
  which of the following statements is true?

- ***ababddababa*** will be tokenized as:
  ~~***ab /abd /d /ababa***~~ ***abab /dd /ababa***
- ***ababdddd*** will be tokenized as: ☑
  abab /dddd
- ***dddabbabab*** will be tokenized as:
  ~~***ddd /a /bbabab***~~ ***ddd /ab /babab***
- ***babad*** will be tokenized as: ☑
  bab /a /d

مدیریت خطا

خطای لغوی

- اگر رشته توسط هیچ ماشین حالتی پذیرفته نشد، یک خطای لغوی رخ داده است.
  - هنگامی که رشته ورودی با هیچ کدام از الگوها همخوانی نداشته باشد، تحلیل گر لغوی نمی تواند جلوتر برود.
  - کامپایلر در هنگام مواجه شدن با اولین خطا متوقف می‌شود.
- خطایاب: خطا را به نحوی مدیریت می‌کند تا تحلیل‌گر بتواند خطا را گزارش داده و در عین حال تحلیل کد را ادامه دهد.

روش‌های مدیریت خطا

- حالت وحشت (Panic Mode): کاراکترهای ورودی آنقدر دور ریخته می‌شوند تا تحلیل‌گر بتواند توکن بعدی را پیدا کند.

- اصلاح خطا: با هدف رفع خطا، متن ورودی را تغییر می‌دهد
  - با انجام کمترین اصلاح در کد به یک کد درست برسیم:
    - یک کاراکتر را از ورودی حذف کند.
    - یک کاراکتر جاافتاده را به ورودی اضافه کند.
    - کاراکتری را با کاراکتر دیگر جایگزین کند.
    - دو کاراکتر کناری را با یکدیگر جابه جا کند.
  - مثال
  - به نظر شما اصلاح کد رو به رو به چه شکلی ممکن است؟
- X = X @ Y;

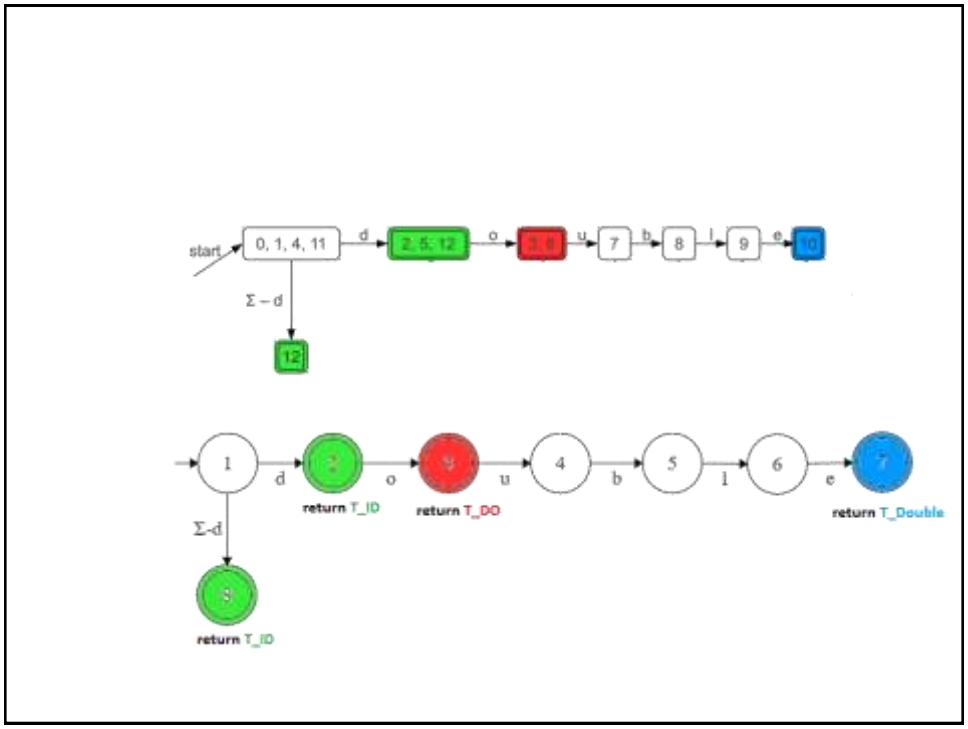
روشهای مدیریت خطا در پویشگر

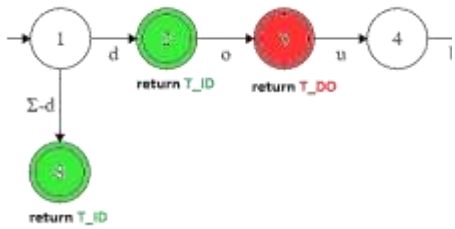
- استراتژی حالت وحشت: کاراکترهای ورودی آنقدر دور ریخته می‌شوند تا به یک توکن قابل قبول برسیم.
  - ایجاد یک توکن خاص با پایین ترین اولویت برای خطا
    - تمام کاراکترها را قبول کند
- استراتژی اصلاح خطا:
  - یک کاراکتر اضافه کند.
  - یک کاراکتر حذف کند.
  - یک کاراکتر را تغییر دهد.
  - یک کاراکتر را جابه‌جا کند.

روش‌های پیاده‌سازی پویشگر

روش یک: پیاده‌سازی مستقیم

- مشخص کردن توکن‌ها با عبارات منظم
- تبدیل عبارات منظم به ماشین حالت
- تبدیل به ماشین DFA
- ایجاد برنامه اجرای DFA با استفاده از جدول انتقال حالت






```
lastToken = null;
bool fail = false;
int state = 1;
while(!fail) {
    char ch = nextChar();
    switch(state) {
        case 1:
            if (ch == 'd') state = 2;
            else state = 8;
            break;
        case 2:
            lastToken = T_ID;
            if (ch == 'o') state = 3;
            else fail = true;
            break;
        case 3:
            lastToken = T_DO;
            if (ch == ‘u') state = 4;
            else fail = true;
            ...
```

روش دوم: ابزارهای تولید خودکار پویشگر

روش دوم:
ابزارهای تولید خودکار پویشگر

- ابزارهای برای تولید کد، بخشهای مختلف کامپایلر مانند Lexer
  - ورودی ابزار: توصیف توکن‌ها (عبارات منظم)
  - خروجی ابزار: کد تولیدشده برای پویشگر
- ANTLRWorks
  - ابزار گرافیکی کار با ANTLR

ANTRL

- ANTLR
  - ANother Tool for Language Recognition
- تولید کد به زبانهای C Java C# Perl Python Ada ,...
- مراحل کار:
  - ایجاد فایل توصیف توکن‌ها
  - تولید فایل کد Lexer با استفاده از ابزار
  - استفاده و اجرای کد تولید شده

1- ANTRL: Specifying Tokens

- زبان استفاده شده در توصیف توکن‌ها عبارت منظم و مشابه با سایر زبان‌های برنامه‌نویسی است.
  - * بستار ستاره
  - + بستار +
  - | یا
  - [a..z] تعریف بازه
  - . (نقطه) به معنای هر کاراکتر است.
  - ؟ یک یا هیچ بار تکرار

1- ANTRL: Specifying Tokens

```
lexer grammar test;
HEX :  '0' ('x' | 'X') ('0'..'9' | 'a'..'f' | 'A'..'F')+ ;
INT :  ('0'..'9')+ ;
ID  :   [a-zA-Z] ([a-zA-Z] | ('0'..'9') | '_' | '.')* ;
WS : [ \r\t\n]+
```

1- ANTRL: Specifying Tokens

- خط اول نام گرامر را مشخص می‌کند.
- خطوط بعد قوانین توصیف توکن‌ها هستند.
  - TOKEN : Regular Expression ;
- نام توکن‌ها باید با حروف بزرگ باشد.
- گرامر در یک فایل ذخیره شود (پسوند معمولا g. است، هرچند اهمیتی ندارد)
- نام فایل و گرامر باید یکی باشد.

2- ANTLR: Running ANTLR

- برای اجرا باید Java روی سیستم نصب باشد.

```
java -jar antlr-4.5.1-complete.jar -Dlanguage=CSharp test.g
```

- Dlanguage: زبان کد تولیدی را مشخص می‌کند
- test.g: نام فایل توصیف توکن‌هاست.
- antlr-4.5.1-complete.jar: فایل اجرای ANTLR است.
  - از آدرس http://www.antlr.org/download.html قابل دانلود است.

3- ANTLR: Using Generated Code

- کد تولید شده توسط کامپایلر زبان درخواست شده، قابل اجراست.
- مثال: تولید و اجرای کد به زبان C#

1. ایجاد یک پروژه جدید در Visual Studio
2. کپی نمودن کلیه فایل‌های تولید شده در مرحله قبل به پروژه
3. ارجاع به کتابخانه Antlr4.Runtime
   - از سایت ANTLR قابل دانلود است.
4. نوشتن سایر کدهای لازم
   - فایل تولید شده فقط lexer است که قادر به تشخیص دنباله توکن‌ها با ورودی دنباله از کاراکترهاست.
     - تابع NextToken از کلاس تولید شده توکن بعدی را از ورودی تشخیص می‌دهد
   - چگونگی استفاده از آن به هدف شما بستگی دارد و باید توسط شما پیاده‌سازی شود.
5. اجرا (کامپایل) پروژه

FLEX

FLEX

- FLEX ابزاری قدیمی اما قوی برای تولید کد بخش LEXER است.
  - ورودی و خروجی آن مشابه ابزار ANTLR است.
  - کد تولید شده تنها به زبان C است.

FLEX: Sample Input

```
%%
[0-9]+                              printf("T_NUMBER");
[0123456789]*[.][0123456789]+       printf("T_FLOAT");
int | double | char                 printf("T_TYPE");
[a-Z_][a-z_0-1]*                    printf("T_ID");
=                                   printf("T_ASSIGN");
[; ]+
.                                   printf("Error");
%%
```

- زبان توصیف توکن‌ها مشابه با ابزار ANTLR است.
- ساختار کلی:

REG Exp	Action
[0123456789]+	printf("T_NUMBER");

- برخلاف ابزار قبل، این ابزار با دیدن هر توکن نوع آن را بر نمی‌گرداند، بلکه دستوری که در بخش دوم داده شده اجرا می‌شود.

پیاده‌سازی یک زبان نمونه

زبان نمونه

- دستورات زبان نمونه:
- تعریف متغیر: نوع متغیر و سپس شناسه متغیر
- نوع فقط می‌تواند Real یا Int باشد.
- مثال
- Real x;
- Int Y;
- محاسبات: انتساب نتیجه یک عبارت در یک متغیر
- عملگرها فقط جمع و تفریق هستند.
- عملوندها ممکن است متغیر یا ثابت عددی باشند.
- مثال
- X=x+2;
- X=y-x+3;

توصیف توکن‌ها

```
T_REAL : 'real';
T_INT : 'int';
REALCONST :   ('0'..'9')* '.' ('0'..'9')+ ;
INTCONST :   ('0'..'9')+ ;
ID  :   [a-zA-Z] ([a-zA-Z] | ('0'..'9') | '_' | '.')* ;
SEPARATOR : ';' ;
ASSIGN : '=';
PLUS : '+';
MIN : '-';
WS : [ \t\r\n]+ ->skip;
```