طراحی کامپایلرها

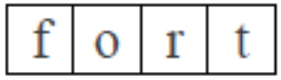
تحلیل لغوی (بخش 2)

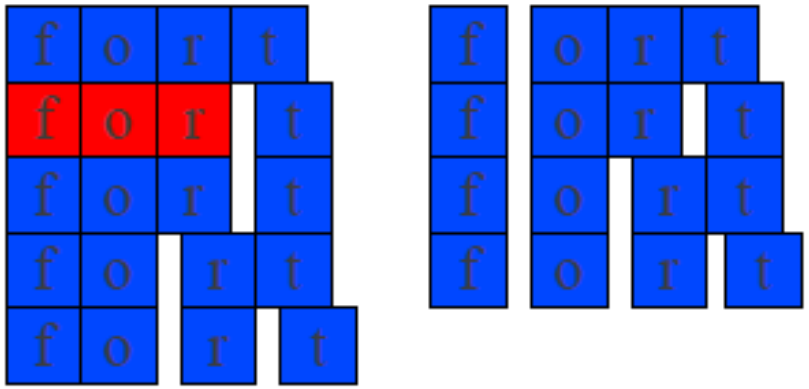
قمرناز تدین تبریزی

چالشها

- چگونه تعیین کنیم کدام لغات با هر توکن مرتبط می شوند؟
- ابهام:

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



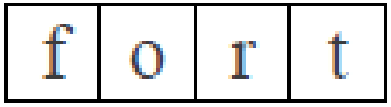


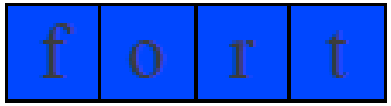
حل مشکل تناقض

- فرض کنید همه توکنها به صورت عبارت باقاعده تعریف شده اند.
- الگوریتم: پویش چپ به راست
- قانون اول: طولانی ترین تطبیق
  - همیشه طولانی ترین پیشوند ممکن از متن باقیمانده تطبیق پیدا می کند.

Lexing Ambiguities

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



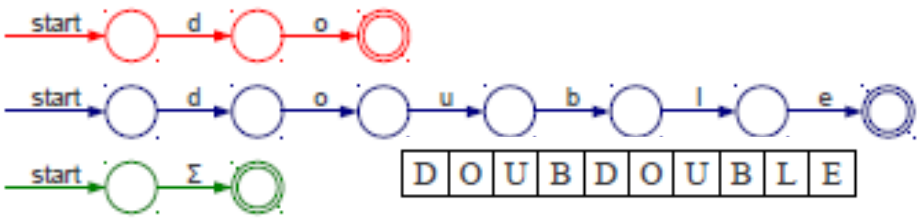


پیاده سازی تطبیق حداکثر

- با داشتن مجموعه ای از عبارتهای باقاعده، چگونه می توانیم از آنها برای پیاده سازی تطبیق حداکثر استفاده کنیم؟
- نظریه:
  - تبدیل عبارت باقاعده به NFA
  - اجرای موازی NFA ها، دانبال کردن تطبیق حداکثر
  - زمانی که همه اتوماتاها متوقف شدند، گزارش آخرین تطبیق و شروع مجدد از آن نقطه

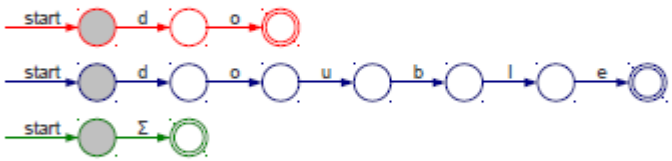
Implementing Maximal Munch

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



Implementing Maximal Munch

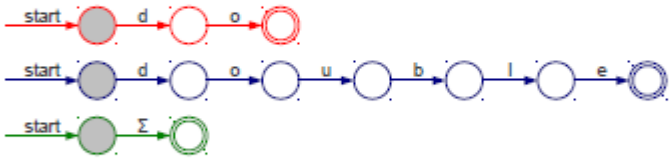
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```

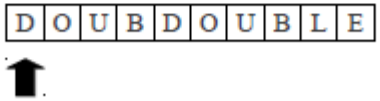


D	O	U	B	D	O	U	B	L	E

Implementing Maximal Munch

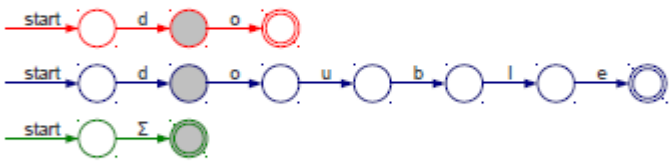
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```





Implementing Maximal Munch

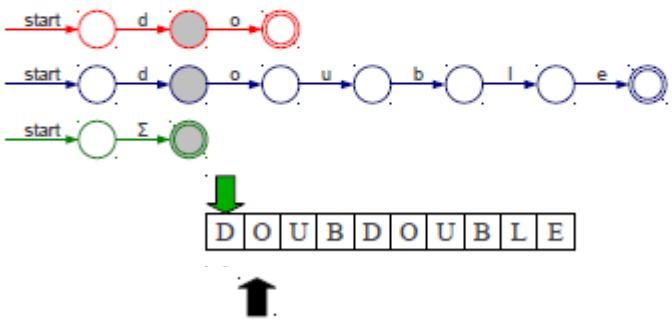
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```



D	O	U	B	D	O	U	B	L	E

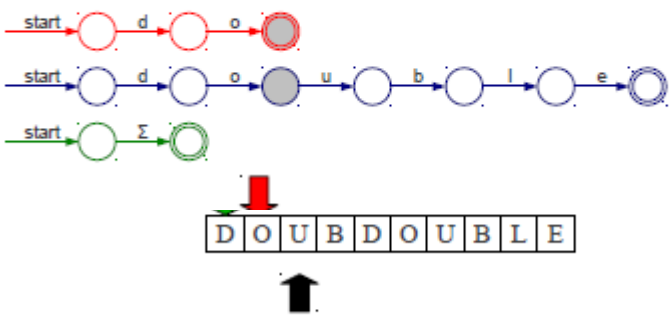
Implementing Maximal Munch

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



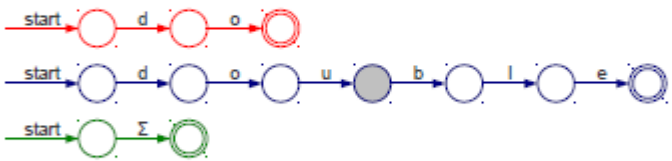
Implementing Maximal Munch

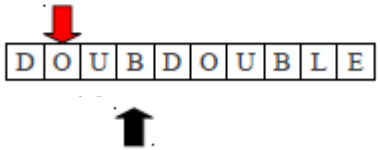
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```



Implementing Maximal Munch

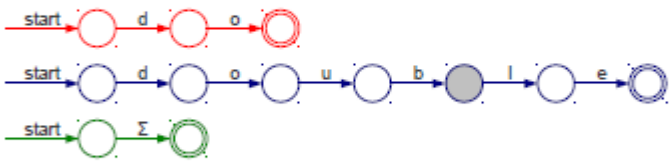
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```

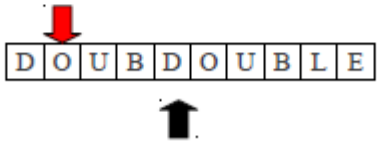




Implementing Maximal Munch

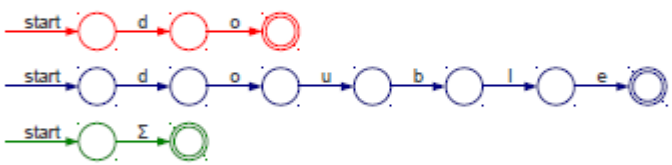
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```

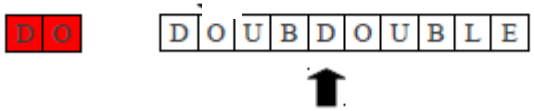




Implementing Maximal Munch

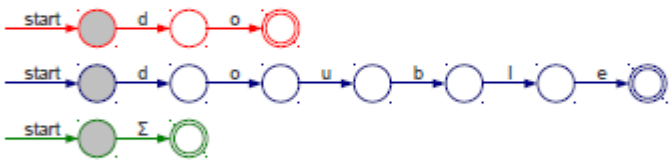
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```

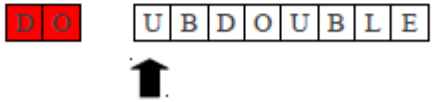




Implementing Maximal Munch

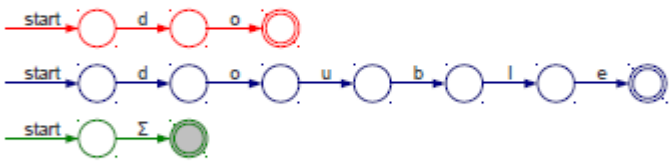
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```

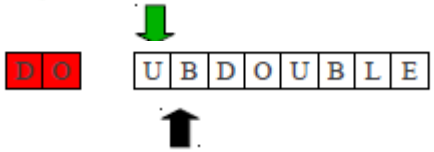




Implementing Maximal Munch

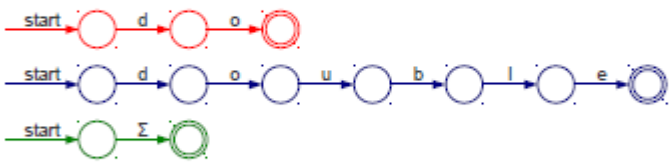
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```

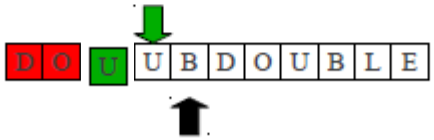




Implementing Maximal Munch

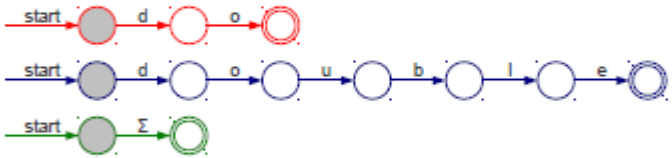
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```

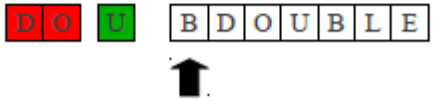




Implementing Maximal Munch

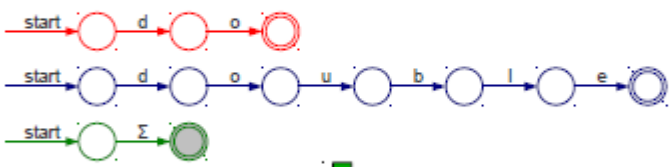
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```

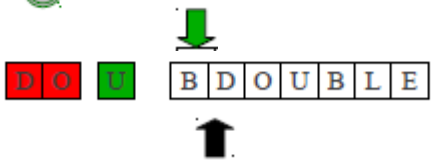




Implementing Maximal Munch

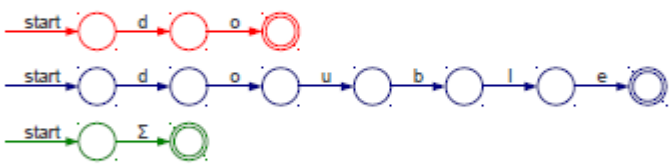
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```

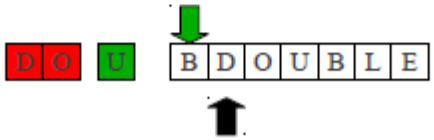




Implementing Maximal Munch

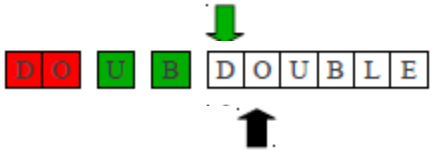
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```





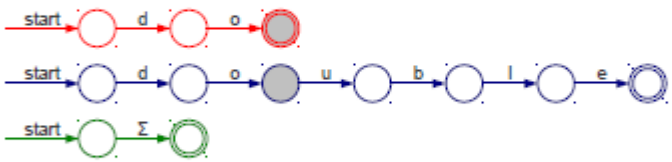
Implementing Maximal Munch

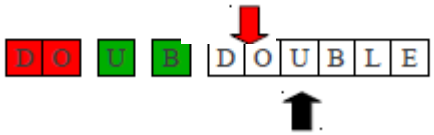
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```



Implementing Maximal Munch

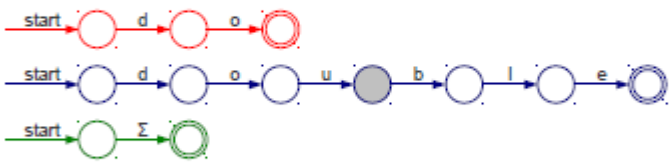
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```

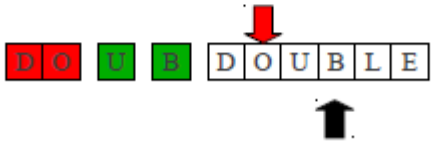




Implementing Maximal Munch

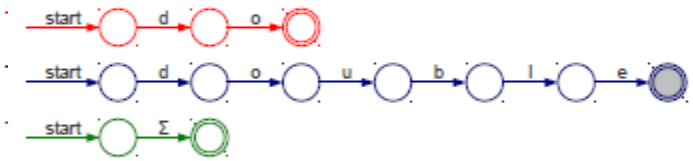
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```





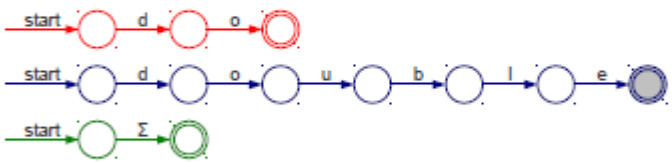
Implementing Maximal Munch

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

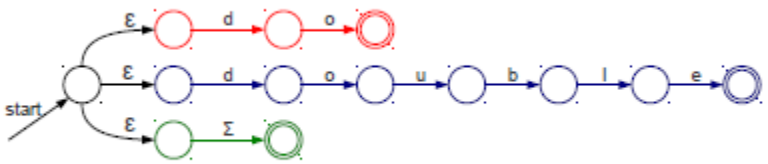


Implementing Maximal Munch

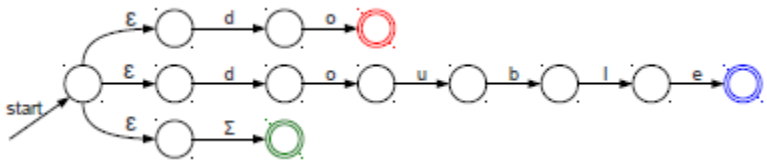
```
T_Do          do
T_Double      double
T_Mystery     [A-Za-z]
```



ساده سازی



یک اتوماتا ساخته می
شوند که همه اتوماتاها را
به صورت موازی اجرا کند



همه حالتهای پذیرش
مشخص می شوند که بر
اساس کدام اتوماتا
طراحی شده اند.

تناقض های دیگر

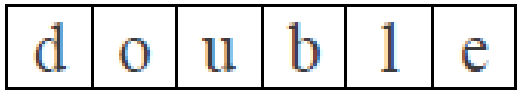

```
T_Do          do
T_Double      double
T_Identifier [A-Za-z_][A-Za-z0-9_]*
```

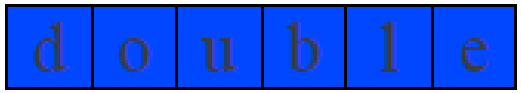
d	o	u	b	l	e

d	o	u	b	l	e
d	o	u	b	l	e

- زمانی که دو عبارت باقاعده منطبق می شوند، عبارتی که اولویت بالاتری دارد انتخاب می شود.
- سیستم اولویت دهی ساده: قانونی که اول تعریف شده انتخاب می شود.

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





- اگرهیچ تطابقی پیدا نشود؟
  - یک قانون اضافه می شود که با همه ورودی ها منطبق می شود و خطا گزارش می شود.

چالشها

- زمانی که چند راه برای پویش ورودی وجود دارد کدام را انتخاب کنیم؟

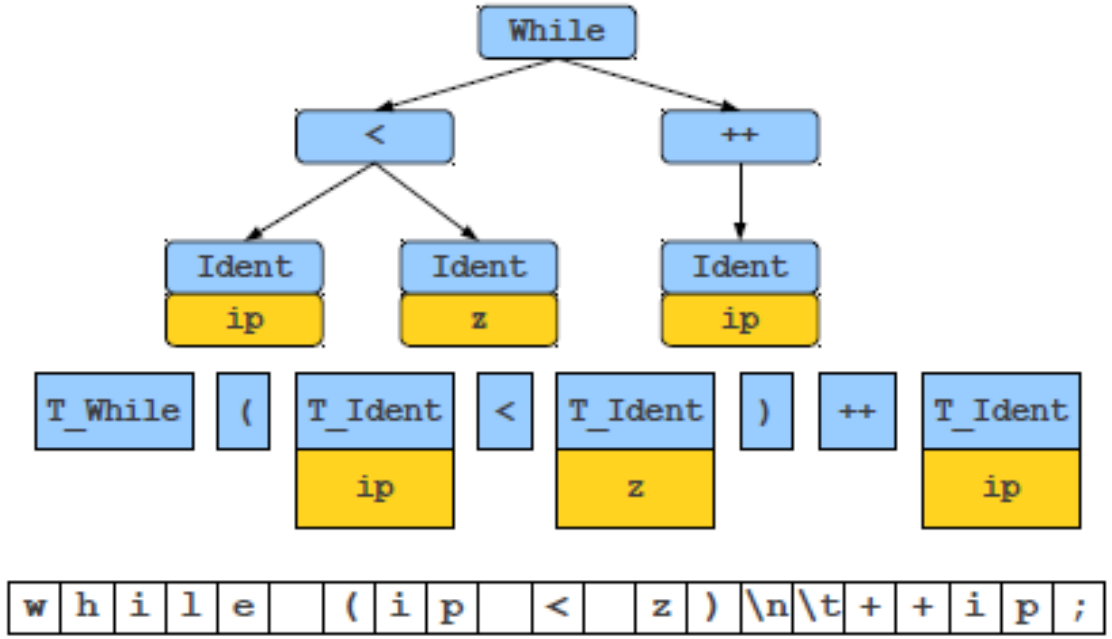
اتوماتای متناهی قطعی (DFA)

- اتوماتاهایی که تا کنون دیدیم NFA بودند.
- DFA مشابه NFA است ولی محدودیتهای بیشتری داردو
  - هر حالت باید برای هر حرف ورودی حداکثر (یا دقیقا) یک انتقال داشته باشد.
  - حرکتهای ε مجاز نیستند.

تغییرات ساختاری

- به جای تعیین این که حالت پذیرش است یا خیر، تعیین می شود کدام نوع توکن منطبق شده است.
- اولویتها در نظر گرفته میشوند.
- DFA زمانی‌گیر میکند که به وضعیتی‌وارد شود که با مجموعه تهی‌مشخص‌شده.
- NFA حافظه کمتر و زمان‌پویش‌بیشتری‌دارد.
- DFA حافظه بیشتر و زمان‌پویش‌کمتری‌دارد.

مثال: زبان Python




```
while (ip < z)
    ++ip;
```

بلاکهای Python

- محدوده لغات با جای خالی مشخص می شود.

```
if w == z:
    a = b
    c = d
else:
    e = f
g = h
```

توکن های جای خالی

- توکنهای خاص اضافه میشوند تا تغییر سطح جلورفتگی را مشخص کند.
- Newline برای تعیین انتهای خط
- Indent: افزایش جلورفتگی متن را مشخص میکند.
- Dedent: کاهش جلورفتگی متن را مشخص میکند.

Scanning Python

```
if w == z: {
    a = b;
    c = d;
} else {
    e = f;
}
g = h;
```