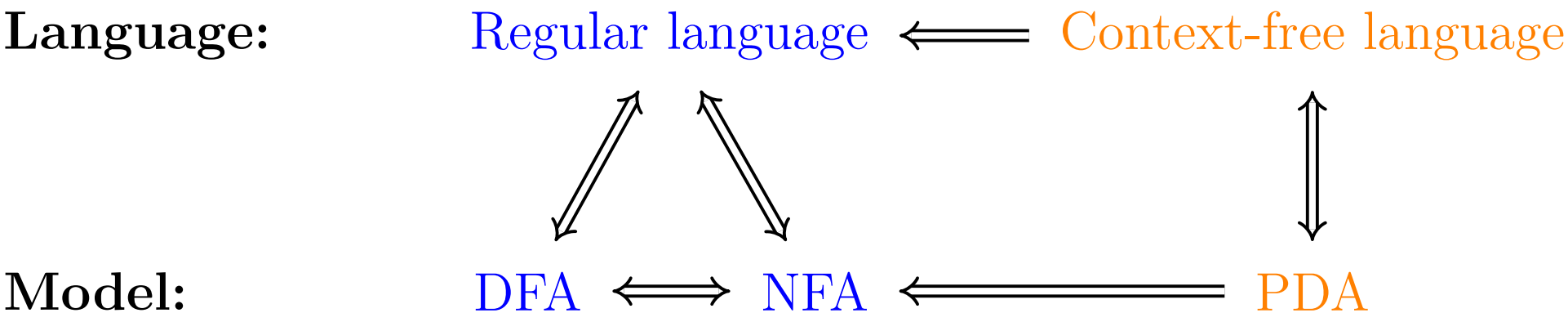
41080 Theory of Computing Science
Week 6 Tutorial Class

Chuanqi Zhang

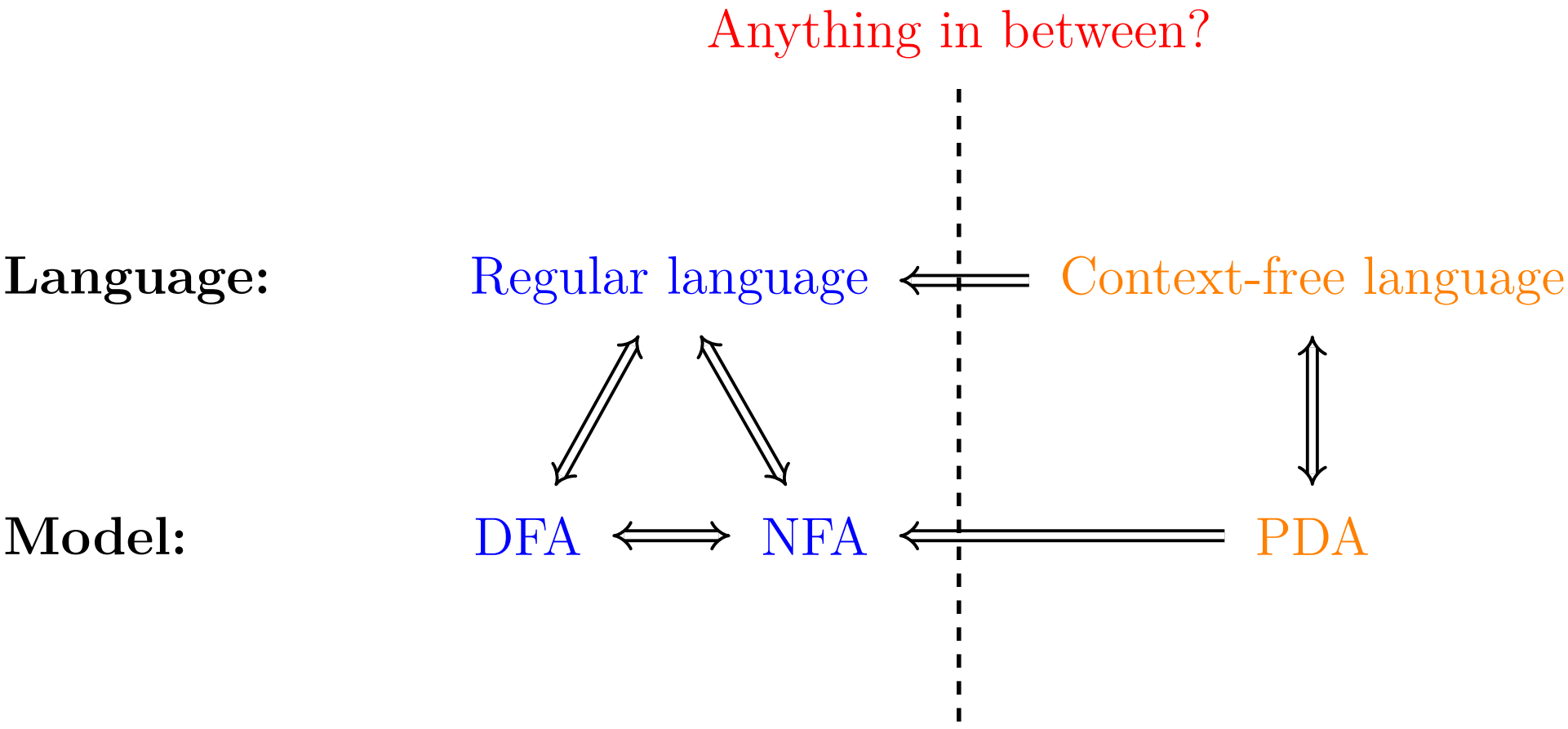
Centre for Quantum Software and Information
University of Technology Sydney

12th September, 2024

Background


Language:
Regular language
Context-free language
Model:
DFA
NFA
PDA

Background


Anything in between?
Language:
Regular language
Context-free language
Model:
DFA
NFA
PDA

Background

- From the practical perspective, we need determinism.
- So people come up with the idea of Deterministic PDA!
- Please find the intuition, definition, etc., on the lecture slides.

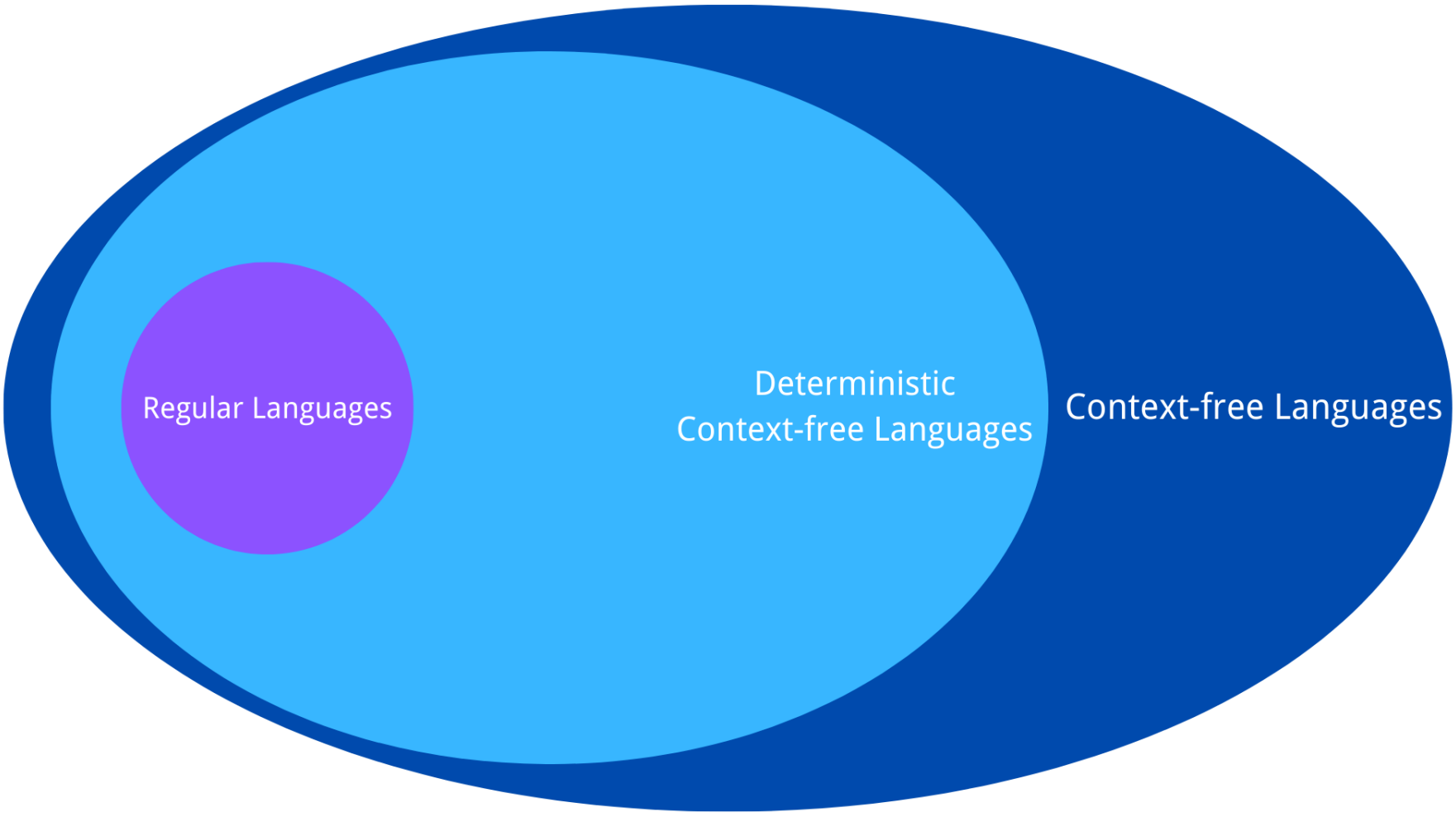
Background

- From the practical perspective, we need determinism.
- So people come up with the idea of Deterministic PDA!
- Please find the intuition, definition, etc., on the lecture slides.

Background

- From the practical perspective, we need determinism.
- So people come up with the idea of Deterministic PDA!
- Please find the intuition, definition, etc., on the lecture slides.

Relations between different languages



Background

From a more practical perspective, we need more restriction.

Problem

Given a grammar $G = (V, \Sigma, R, S)$ and a string $w \in L(G)$, how can we construct the process of generating w from S?

This is achieved through a parser. Two possible approaches:

- Top-down parsers: start from S and derive the string w.
- Bottom-up parsers: start from the string w and reach S.

Background

From a more practical perspective, we need more restriction.

Problem

Given a grammar $G = (V, \Sigma, R, S)$ and a string $w \in L(G)$, how can we construct the process of generating w from S?

This is achieved through a parser. Two possible approaches:

- Top-down parsers: start from S and derive the string w.
- Bottom-up parsers: start from the string w and reach S.

Background

From a more practical perspective, we need more restriction.

Problem

Given a grammar $G = (V, \Sigma, R, S)$ and a string $w \in L(G)$, how can we construct the process of generating w from S?

This is achieved through a parser. Two possible approaches:

- Top-down parsers: start from S and derive the string w.
- Bottom-up parsers: start from the string w and reach S.

Background

From a more practical perspective, we need more restriction.

Problem

Given a grammar $G = (V, \Sigma, R, S)$ and a string $w \in L(G)$, how can we construct the process of generating w from S?

This is achieved through a parser. Two possible approaches:

- Top-down parsers: start from S and derive the string w.
- Bottom-up parsers: start from the string w and reach S.

Background

From a more practical perspective, we need more restriction.

Problem

Given a grammar $G = (V, \Sigma, R, S)$ and a string $w \in L(G)$, how can we construct the process of generating w from S?

This is achieved through a parser. Two possible approaches:

- Top-down parsers: start from S and derive the string w.
- Bottom-up parsers: start from the string w and reach S.

Background

From a more practical perspective, we need more restriction.

Problem

Given a grammar $G = (V, \Sigma, R, S)$ and a string $w \in L(G)$, how can we construct the process of generating w from S?

This is achieved through a parser. Two possible approaches:

- Top-down parsers: start from S and derive the string w.
- Bottom-up parsers: start from the string w and reach S.

Background

From a more practical perspective, we need more restriction.

Problem

Given a grammar $G = (V, \Sigma, R, S)$ and a string $w \in L(G)$, how can we construct the process of generating w from S?

This is achieved through a parser. Two possible approaches:

- Top-down parsers: start from S and derive the string w.
- Bottom-up parsers: start from the string w and reach S.

Why is it called top-down?

For some parsers, we can build a parsing tree to illustrate the generation process.

Example (Top-down parsers)

How string id + id is generated from the start variable E by applying a bunch of grammar rules:

This particular example is not only top-down, but also

Why is it called top-down?

For some parsers, we can build a parsing tree to illustrate the generation process.

Example (Top-down parsers)

How string id + id is generated from the start variable E by applying a bunch of grammar rules:

This particular example is not only top-down, but also

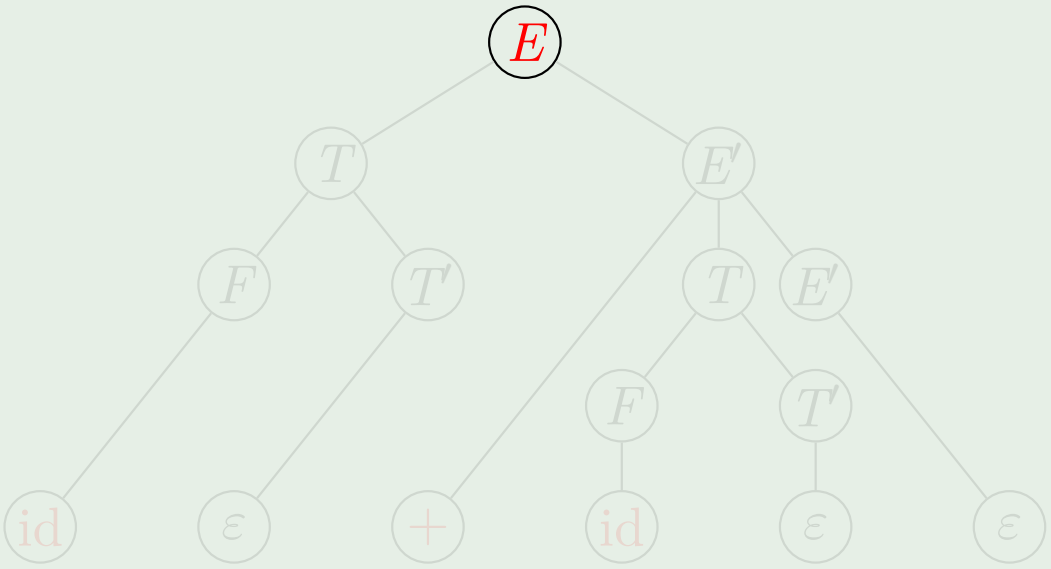
1. left-right;
2. left-most derivation;
3. one symbol look-ahead.

Why is it called top-down?

For some parsers, we can build a parsing tree to illustrate the generation process.

Example (Top-down parsers)

How string id + id is generated from the start variable E by applying a bunch of grammar rules:



This particular example is not only top-down, but also

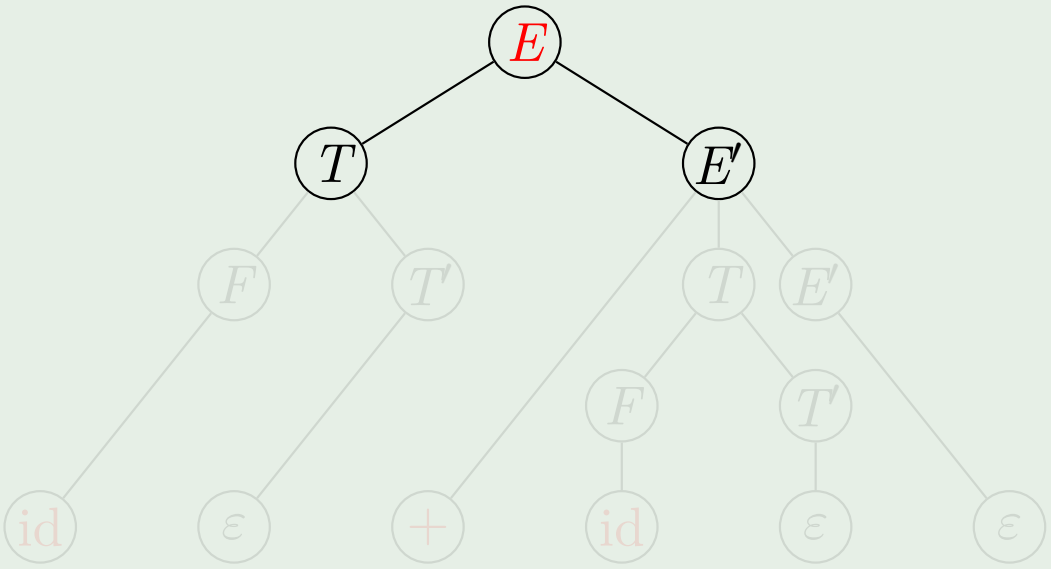
1. left-right;
2. left-most derivation;
3. one symbol look-ahead.

Why is it called top-down?

For some parsers, we can build a parsing tree to illustrate the generation process.

Example (Top-down parsers)

How string id + id is generated from the start variable E by applying a bunch of grammar rules:



This particular example is not only top-down, but also

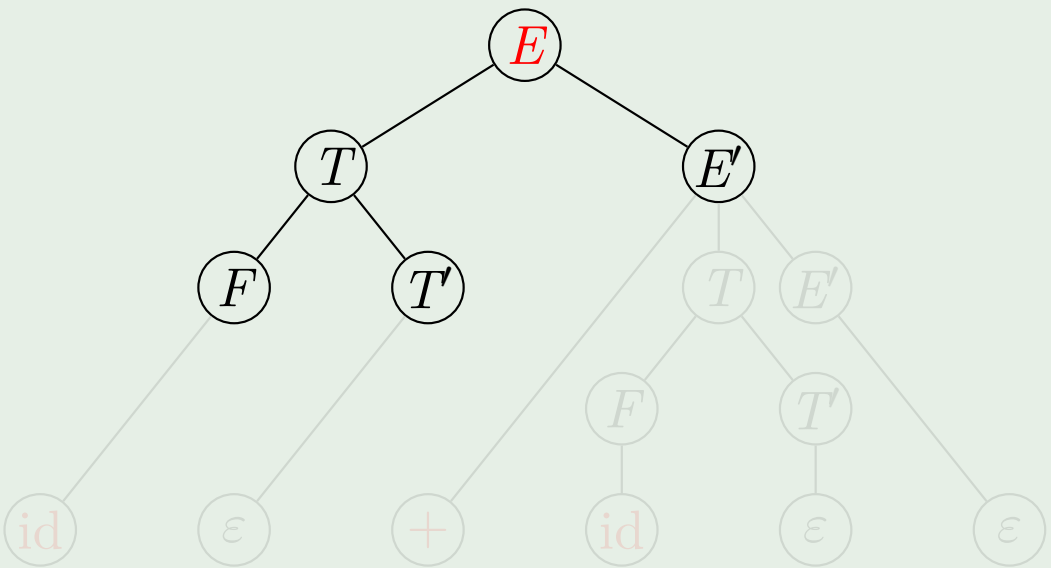
1. left-right;
2. left-most derivation;
3. one symbol look-ahead.

Why is it called top-down?

For some parsers, we can build a parsing tree to illustrate the generation process.

Example (Top-down parsers)

How string id + id is generated from the start variable E by applying a bunch of grammar rules:



This particular example is not only top-down, but also

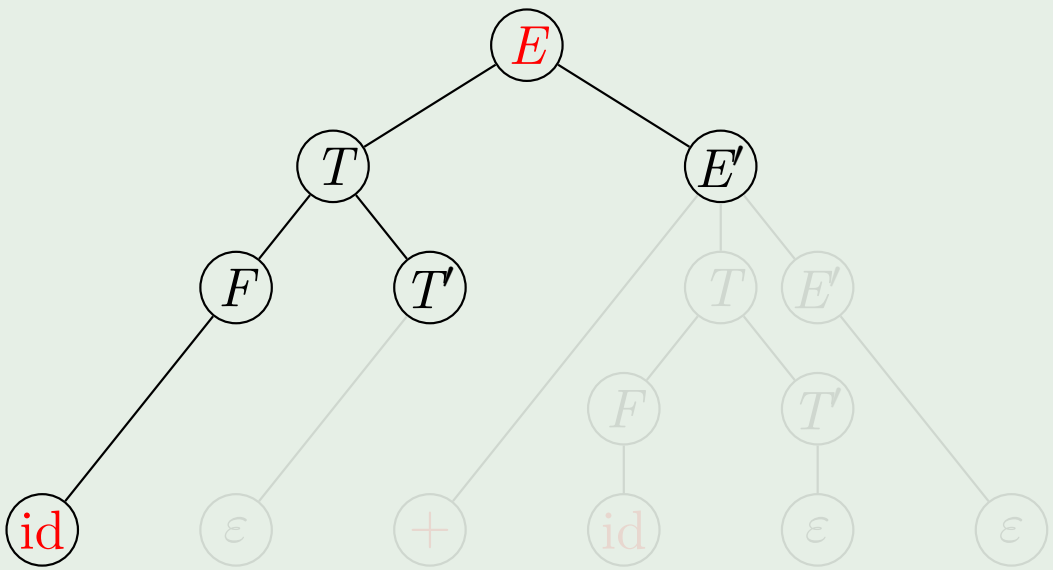
1. left-right;
2. left-most derivation;
3. one symbol look-ahead.

Why is it called top-down?

For some parsers, we can build a parsing tree to illustrate the generation process.

Example (Top-down parsers)

How string id + id is generated from the start variable E by applying a bunch of grammar rules:



This particular example is not only top-down, but also

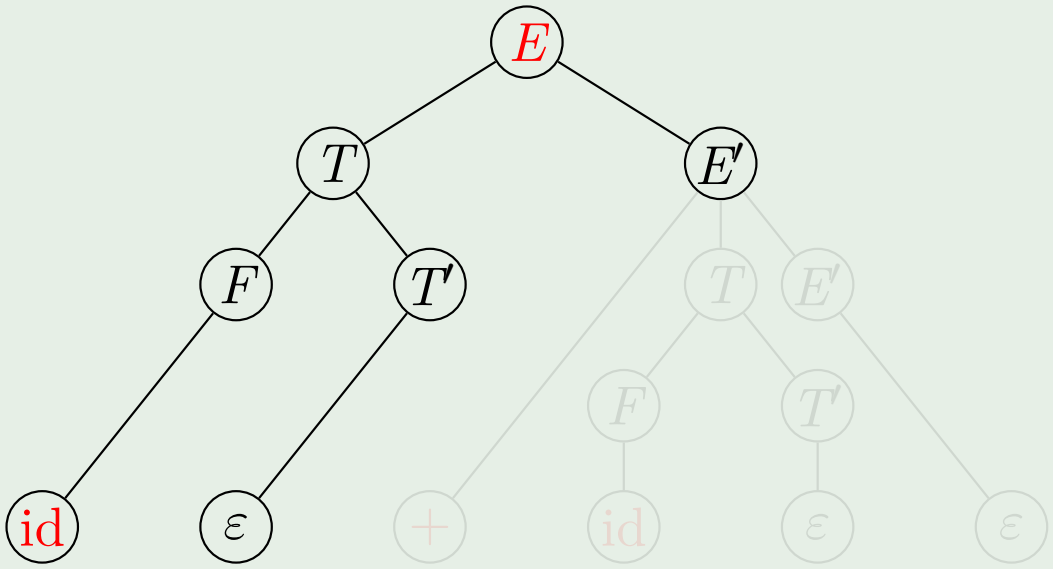
1. left-right;
2. left-most derivation;
3. one symbol look-ahead.

Why is it called top-down?

For some parsers, we can build a parsing tree to illustrate the generation process.

Example (Top-down parsers)

How string id + id is generated from the start variable E by applying a bunch of grammar rules:



This particular example is not only top-down, but also

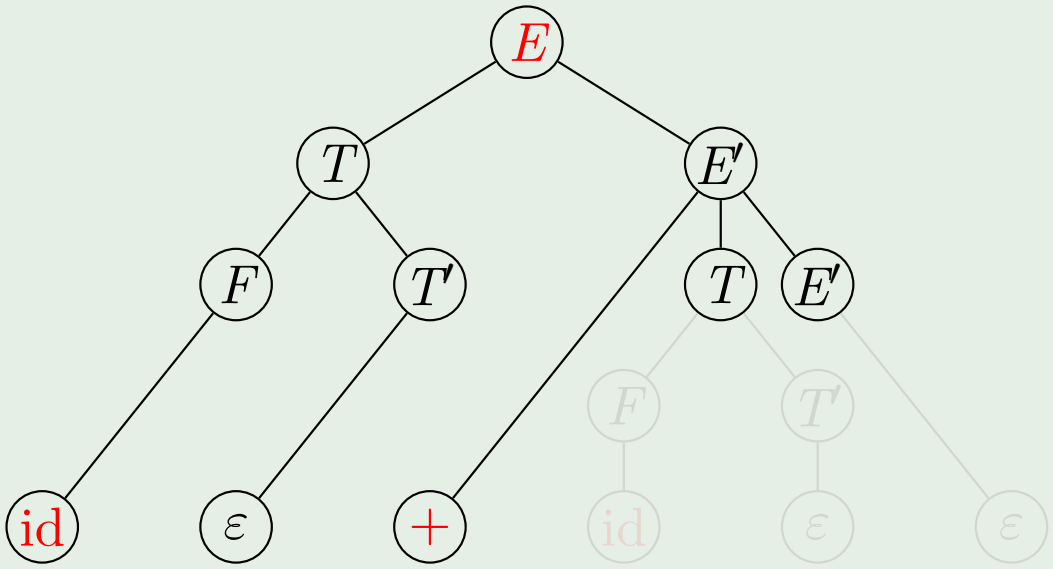
1. left-right;
2. left-most derivation;
3. one symbol look-ahead.

Why is it called top-down?

For some parsers, we can build a parsing tree to illustrate the generation process.

Example (Top-down parsers)

How string $\mathrm{id} + \mathrm{id}$ is generated from the start variable E by applying a bunch of grammar rules:



This particular example is not only top-down, but also

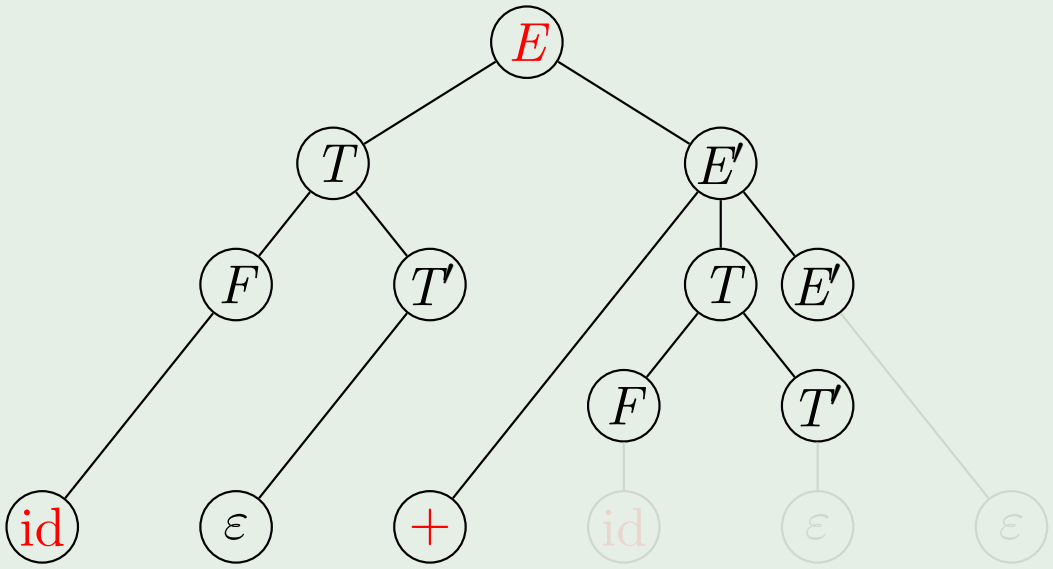
1. left-right;
2. left-most derivation;
3. one symbol look-ahead.

Why is it called top-down?

For some parsers, we can build a parsing tree to illustrate the generation process.

Example (Top-down parsers)

How string id + id is generated from the start variable E by applying a bunch of grammar rules:



This particular example is not only top-down, but also

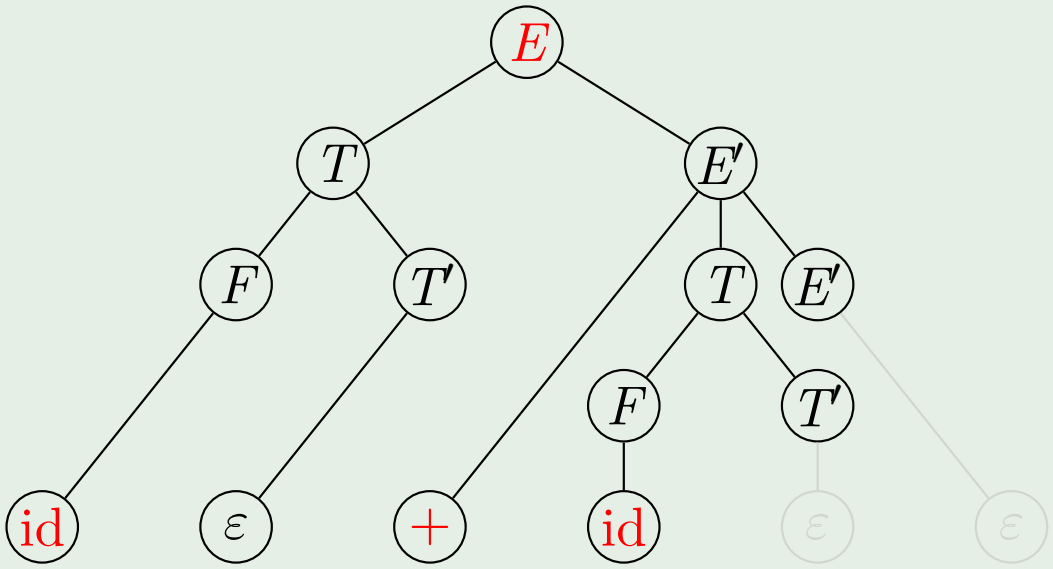
1. left-right;
2. left-most derivation;
3. one symbol look-ahead.

Why is it called top-down?

For some parsers, we can build a parsing tree to illustrate the generation process.

Example (Top-down parsers)

How string $\mathrm{id} + \mathrm{id}$ is generated from the start variable E by applying a bunch of grammar rules:



This particular example is not only top-down, but also

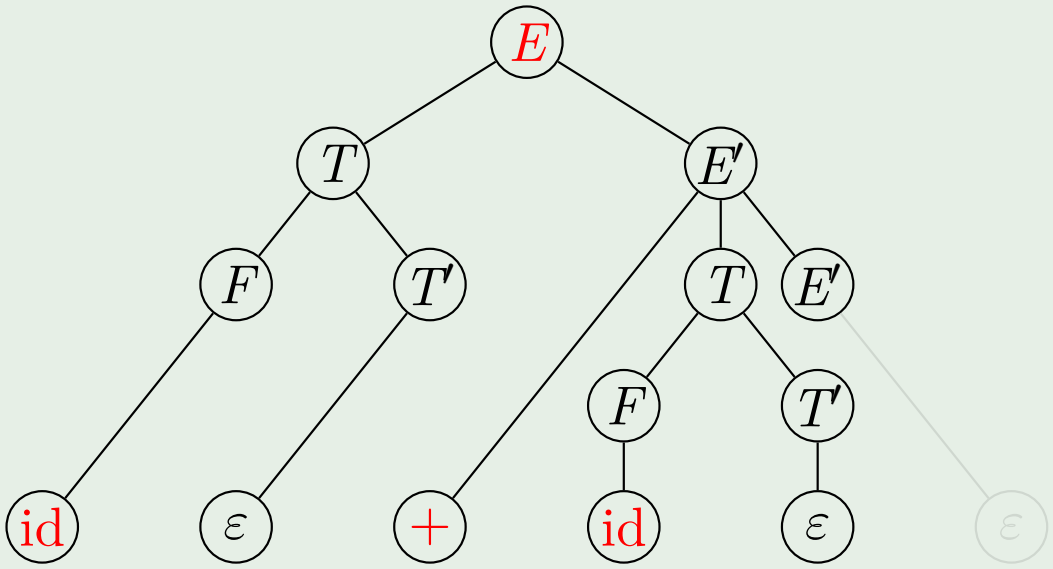
1. left-right;
2. left-most derivation;
3. one symbol look-ahead.

Why is it called top-down?

For some parsers, we can build a parsing tree to illustrate the generation process.

Example (Top-down parsers)

How string $\mathrm{id} + \mathrm{id}$ is generated from the start variable E by applying a bunch of grammar rules:



This particular example is not only top-down, but also

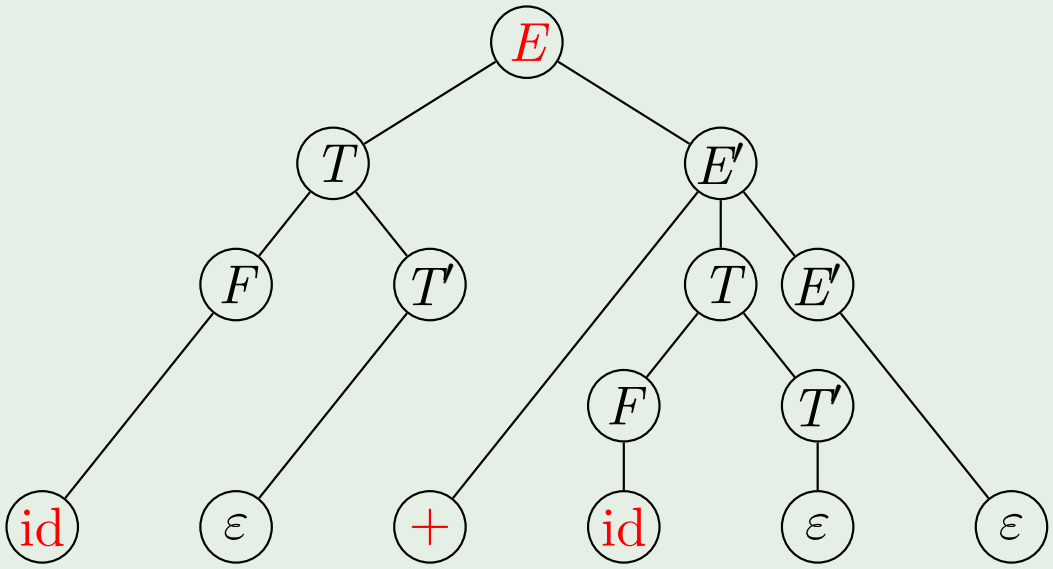
1. left-right;
2. left-most derivation;
3. one symbol look-ahead.

Why is it called top-down?

For some parsers, we can build a parsing tree to illustrate the generation process.

Example (Top-down parsers)

How string $\mathrm{id} + \mathrm{id}$ is generated from the start variable E by applying a bunch of grammar rules:



This particular example is not only top-down, but also

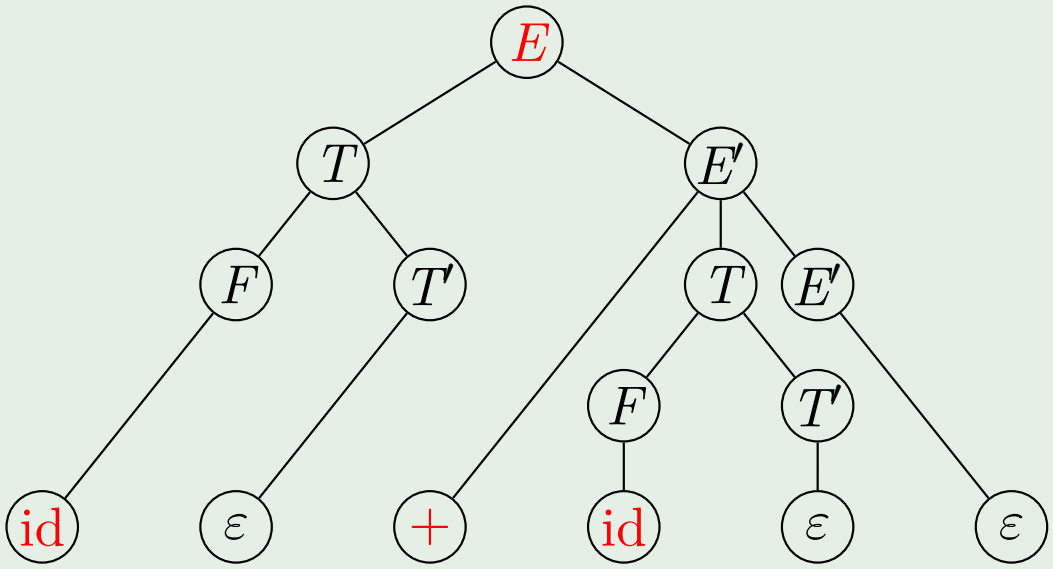
1. left-right;
2. left-most derivation;
3. one symbol look-ahead.

Why is it called top-down?

For some parsers, we can build a parsing tree to illustrate the generation process.

Example (Top-down parsers)

How string id + id is generated from the start variable E by applying a bunch of grammar rules:



This particular example is not only top-down, but also

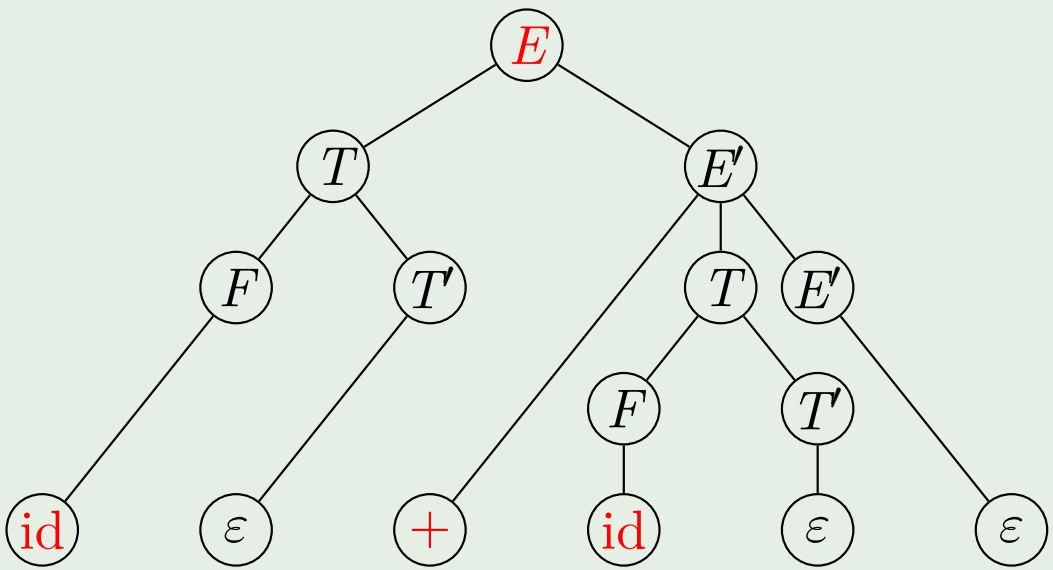
1. left-right;
2. left-most derivation;
3. one symbol look-ahead.

Why is it called top-down?

For some parsers, we can build a parsing tree to illustrate the generation process.

Example (Top-down parsers)

How string id + id is generated from the start variable E by applying a bunch of grammar rules:



This particular example is not only top-down, but also

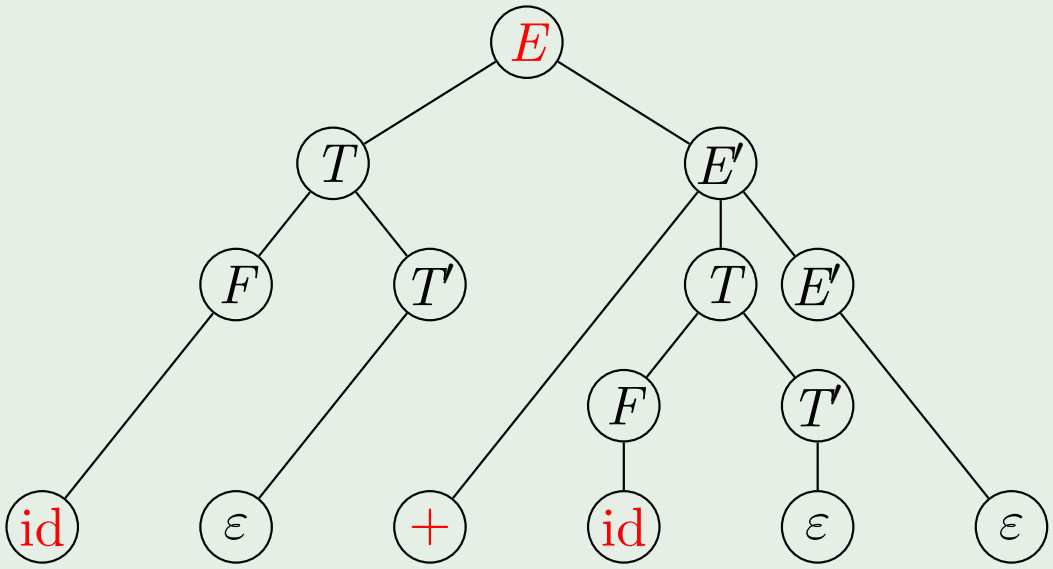
1. left-right;
2. left-most derivation;
3. one symbol look-ahead.

Why is it called top-down?

For some parsers, we can build a parsing tree to illustrate the generation process.

Example (Top-down parsers)

How string id + id is generated from the start variable E by applying a bunch of grammar rules:



This particular example is not only top-down, but also

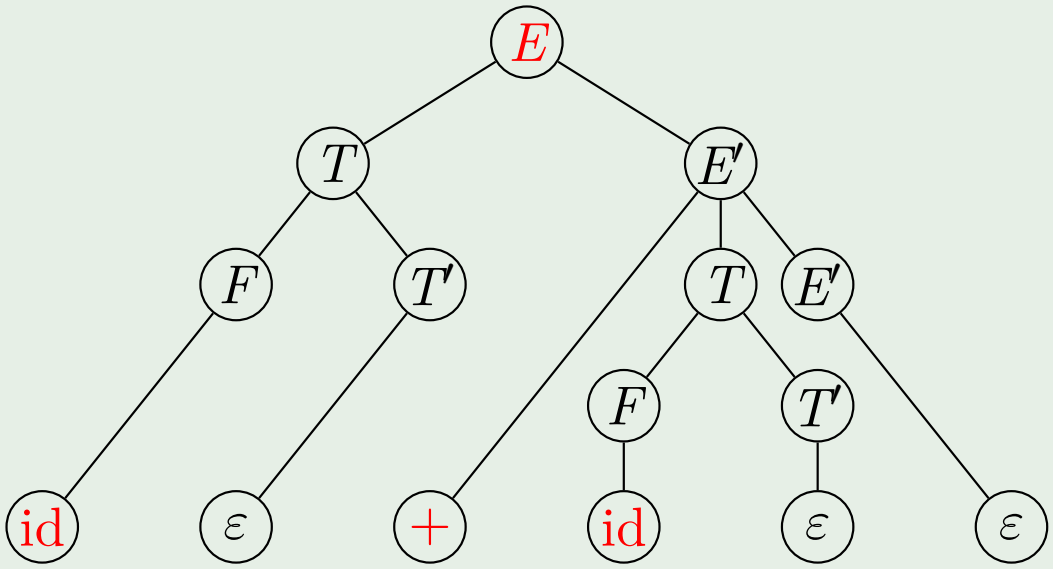
1. left-right;
2. left-most derivation;
3. one symbol look-ahead.

Why is it called top-down?

For some parsers, we can build a parsing tree to illustrate the generation process.

Example (Top-down parsers)

How string id + id is generated from the start variable E by applying a bunch of grammar rules:



This particular example is not only top-down, but also

1. left-right;
2. left-most derivation;
3. one symbol look-ahead.

$LL(1)$ parsers

- We focus on $LL(1)$ parsers, a special type of top-down parsers.
  - L: left-right
  - L: left-most derivation
  - 1: one symbol look-ahead
- Not every grammar allows for $LL(1)$ parsing.
- Some grammars can be converted to allow for $LL(1)$ parsing by the techniques called left recursion removal and left factoring. Please refer to the lecture slides.

$LL(1)$ parsers

- We focus on $LL(1)$ parsers, a special type of top-down parsers.
  - L: left-right
  - L: left-most derivation
  - 1: one symbol look-ahead
- Not every grammar allows for $LL(1)$ parsing.
- Some grammars can be converted to allow for $LL(1)$ parsing by the techniques called left recursion removal and left factoring. Please refer to the lecture slides.

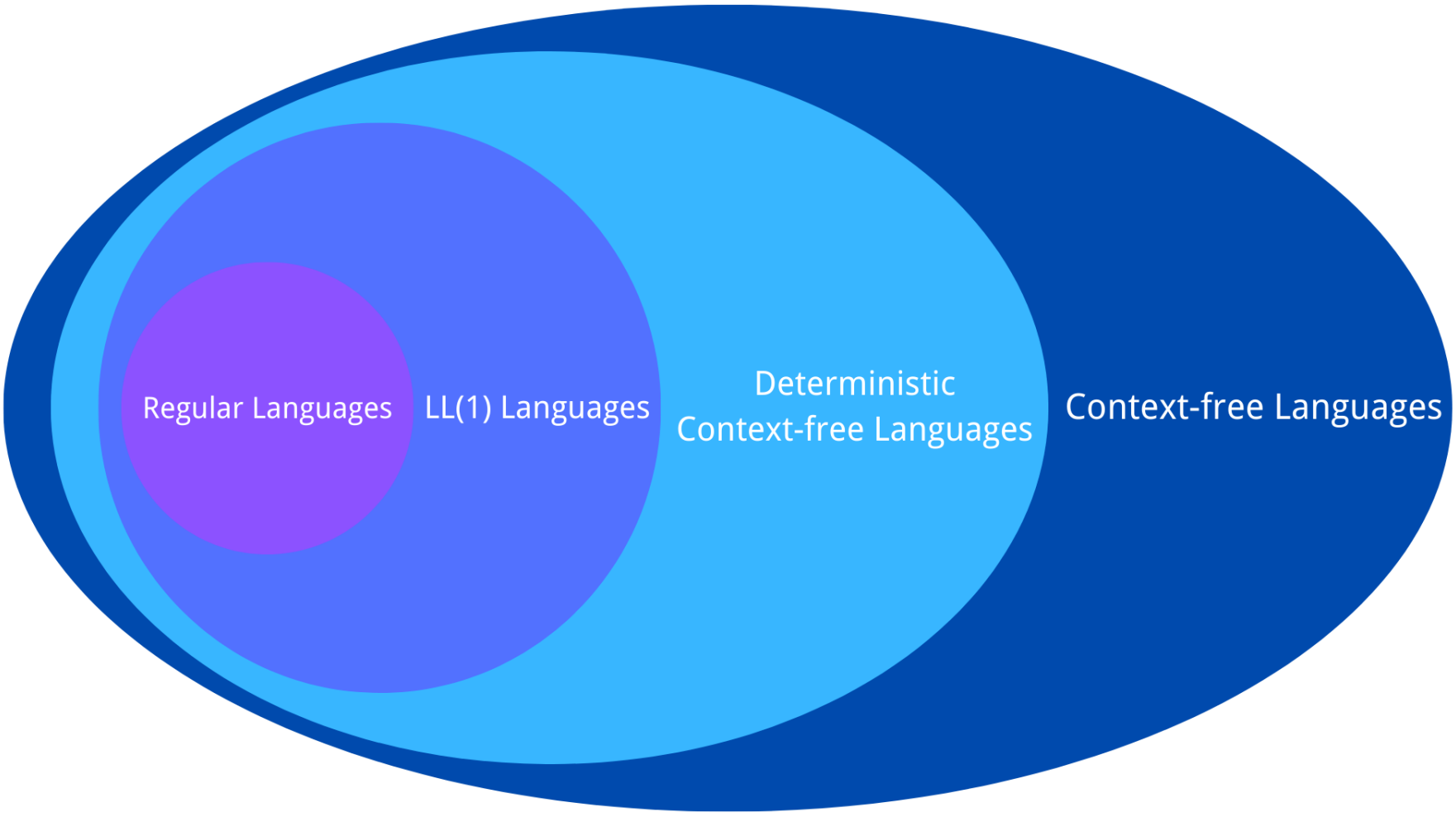
$LL(1)$ parsers

- We focus on $LL(1)$ parsers, a special type of top-down parsers.
  - L: left-right
  - L: left-most derivation
  - 1: one symbol look-ahead
- Not every grammar allows for $LL(1)$ parsing.
- Some grammars can be converted to allow for $LL(1)$ parsing by the techniques called left recursion removal and left factoring. Please refer to the lecture slides.

$LL(1)$ parsers

- We focus on $LL(1)$ parsers, a special type of top-down parsers.
  - L: left-right
  - L: left-most derivation
  - 1: one symbol look-ahead
- Not every grammar allows for $LL(1)$ parsing.
- Some grammars can be converted to allow for $LL(1)$ parsing by the techniques called left recursion removal and left factoring. Please refer to the lecture slides.

Relations between different languages



What is a predicative parsing table?

Generate string id + id from the start variable E:

Example

1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \text{id}$

	id	+	*	(	)	$
E	1			1		
E'		2			3	3
T	4			4		
T'		6	5		6	6
F	8			7		

Predicative parsing table for the left-hand side grammar.

What is a predicative parsing table?

Generate string id + id from the start variable E:

$$E \overset{1}{\Rightarrow} TE'$$

Example

1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \text{id}$

	id	+	*	(	)	$
E	1			1		
E'		2			3	3
T	4			4		
T'		6	5		6	6
F	8			7		

Predicative parsing table for the left-hand side grammar.

What is a predicative parsing table?

Generate string id + id from the start variable E:

$$E \overset{1}{\Rightarrow} TE' \overset{4}{\Rightarrow} FT'E' \overset{8}{\Rightarrow} \mathrm{id}T'E' \overset{6}{\Rightarrow} \mathrm{id}E' \overset{2}{\Rightarrow} \mathrm{id} + TE' \Rightarrow \cdots \Rightarrow \mathrm{id} + \mathrm{id}$$

Example

1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \mathrm{id}$

	id	+	*	(	)	$
E	1			1		
E'		2			3	3
T	4			4		
T'		6	5		6	6
F	8			7		

Predicative parsing table for the left-hand side grammar.

What is a predicative parsing table?

Generate string id + id from the start variable E:

$$E \overset{1}{\Rightarrow} TE' \overset{4}{\Rightarrow} FT'E' \overset{8}{\Rightarrow} \mathrm{id}\,T'E' \overset{6}{\Rightarrow} \mathrm{id}\,E' \overset{2}{\Rightarrow} \mathrm{id} + TE' \Rightarrow \cdots \Rightarrow \mathrm{id} + \mathrm{id}$$

Example

1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \mathrm{id}$

	id	+	*	(	)	$
E	1			1		
E'		2			3	3
T	4			4		
T'		6	5		6	6
F	8			7		

Predicative parsing table for the left-hand side grammar.

What is a predicative parsing table?

Generate string id + id from the start variable E:

$$E \overset{1}{\Rightarrow} TE' \overset{4}{\Rightarrow} FT'E' \overset{8}{\Rightarrow} \mathrm{id}\,T'E' \overset{6}{\Rightarrow} \mathrm{id}\,E' \overset{2}{\Rightarrow} \mathrm{id} + TE' \Rightarrow \cdots \Rightarrow \mathrm{id} + \mathrm{id}$$

Example

1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \mathrm{id}$

	id	+	*	(	)	$
E	1			1		
E'		2			3	3
T	4			4		
T'		6	5		6	6
F	8			7		

Predicative parsing table for the left-hand side grammar.

What is a predicative parsing table?

Generate string id + id from the left-most variable T:

$$E \overset{1}{\Rightarrow} TE' \overset{4}{\Rightarrow} FT'E' \overset{8}{\Rightarrow} \mathrm{id}\,T'E' \overset{6}{\Rightarrow} \mathrm{id}\,E' \overset{2}{\Rightarrow} \mathrm{id} + TE' \Rightarrow \cdots \Rightarrow \mathrm{id} + \mathrm{id}$$

Example

1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \mathrm{id}$

	id	+	*	(	)	$
E	1			1		
E'		2			3	3
T	4			4		
T'		6	5		6	6
F	8			7		

Predicative parsing table for the left-hand side grammar.

What is a predicative parsing table?

Generate string id + id from the left-most variable T:

$$E \overset{1}{\Rightarrow} TE' \overset{4}{\Rightarrow} FT'E' \overset{8}{\Rightarrow} \mathrm{id}\,T'E' \overset{6}{\Rightarrow} \mathrm{id}\,E' \overset{2}{\Rightarrow} \mathrm{id} + TE' \Rightarrow \cdots \Rightarrow \mathrm{id} + \mathrm{id}$$

Example

1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \mathrm{id}$

	id	+	*	(	)	$
E	1			1		
E'		2			3	3
T	4			4		
T'		6	5		6	6
F	8			7		

Predicative parsing table for the left-hand side grammar.

What is a predicative parsing table?

Generate string id + id from the left-most variable T:

$$E \overset{1}{\Rightarrow} TE' \overset{4}{\Rightarrow} FT'E' \overset{8}{\Rightarrow} \mathrm{id}\,T'E' \overset{6}{\Rightarrow} \mathrm{id}\,E' \overset{2}{\Rightarrow} \mathrm{id} + TE' \Rightarrow \cdots \Rightarrow \mathrm{id} + \mathrm{id}$$

Example

1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \mathrm{id}$

	id	+	*	(	)	$
E	1			1		
E'		2			3	3
T	4			4		
T'		6	5		6	6
F	8			7		

Predicative parsing table for the left-hand side grammar.

What is a predicative parsing table?

Generate string id + id from the left-most variable F:

$$E \overset{1}{\Rightarrow} TE' \overset{4}{\Rightarrow} FT'E' \overset{8}{\Rightarrow} \mathrm{id}\,T'E' \overset{6}{\Rightarrow} \mathrm{id}\,E' \overset{2}{\Rightarrow} \mathrm{id} + TE' \Rightarrow \cdots \Rightarrow \mathrm{id} + \mathrm{id}$$

Example

1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \mathrm{id}$

	id	+	*	(	)	$
E	1			1		
E'		2			3	3
T	4			4		
T'		6	5		6	6
F	8			7		

Predicative parsing table for the left-hand side grammar.

What is a predicative parsing table?

Generate string id + id from the left-most variable F:

$$E \overset{1}{\Rightarrow} TE' \overset{4}{\Rightarrow} FT'E' \overset{8}{\Rightarrow} \mathrm{id}\,T'E' \overset{6}{\Rightarrow} \mathrm{id}\,E' \overset{2}{\Rightarrow} \mathrm{id} + TE' \Rightarrow \cdots \Rightarrow \mathrm{id} + \mathrm{id}$$

Example

1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \mathrm{id}$

	id	+	*	(	)	$
E	1			1		
E'		2			3	3
T	4			4		
T'		6	5		6	6
F	8			7		

Predicative parsing table for the left-hand side grammar.

What is a predicative parsing table?

Generate string id + id from the left-most variable F:

$$E \overset{1}{\Rightarrow} TE' \overset{4}{\Rightarrow} FT'E' \overset{8}{\Rightarrow} \mathrm{id}\,T'E' \overset{6}{\Rightarrow} \mathrm{id}\,E' \overset{2}{\Rightarrow} \mathrm{id} + TE' \Rightarrow \cdots \Rightarrow \mathrm{id} + \mathrm{id}$$

Example

1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \mathrm{id}$

	id	+	*	(	)	$
E	1			1		
E'		2			3	3
T	4			4		
T'		6	5		6	6
F	8			7		

Predicative parsing table for the left-hand side grammar.

What is a predicative parsing table?

Generate string id + id from the left-most variable T':

$$E \overset{1}{\Rightarrow} TE' \overset{4}{\Rightarrow} FT'E' \overset{8}{\Rightarrow} \mathrm{id}\,T'E' \overset{6}{\Rightarrow} \mathrm{id}\,E' \overset{2}{\Rightarrow} \mathrm{id} + TE' \Rightarrow \cdots \Rightarrow \mathrm{id} + \mathrm{id}$$

Example

1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \mathrm{id}$

	id	+	*	(	)	$
E	1			1		
E'		2			3	3
T	4			4		
T'		6	5		6	6
F	8			7		

Predicative parsing table for the left-hand side grammar.

UTS:QSI
CENTRE FOR QUANTUM SOFTWARE AND INFORMATION

What is a predicative parsing table?

Generate string id + id from the left-most variable T':

$$E \overset{1}{\Rightarrow} TE' \overset{4}{\Rightarrow} FT'E' \overset{8}{\Rightarrow} \mathrm{id}\,T'E' \overset{6}{\Rightarrow} \mathrm{id}\,E' \overset{2}{\Rightarrow} \mathrm{id} + TE' \Rightarrow \cdots \Rightarrow \mathrm{id} + \mathrm{id}$$

Example

1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \mathrm{id}$

	id	+	*	(	)	$
E	1			1		
E'		2			3	3
T	4			4		
T'		6	5		6	6
F	8			7		

Predicative parsing table for the left-hand side grammar.

What is a predicative parsing table?

Generate string id + id from the left-most variable T':

$$E \overset{1}{\Rightarrow} TE' \overset{4}{\Rightarrow} FT'E' \overset{8}{\Rightarrow} \mathrm{id}\,T'E' \overset{6}{\Rightarrow} \mathrm{id}\,E' \overset{2}{\Rightarrow} \mathrm{id} + TE' \Rightarrow \cdots \Rightarrow \mathrm{id} + \mathrm{id}$$

Example

1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \mathrm{id}$

	id	+	*	(	)	$
E	1			1		
E'		2			3	3
T	4			4		
T'		6	5		6	6
F	8			7		

Predicative parsing table for the left-hand side grammar.

What is a predicative parsing table?

Generate string id + id from the left-most variable E':

$$E \overset{1}{\Rightarrow} TE' \overset{4}{\Rightarrow} FT'E' \overset{8}{\Rightarrow} \mathrm{id}\,T'E' \overset{6}{\Rightarrow} \mathrm{id}\,E' \overset{2}{\Rightarrow} \mathrm{id} + TE' \Rightarrow \cdots \Rightarrow \mathrm{id} + \mathrm{id}$$

Example

1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \mathrm{id}$

	id	+	*	(	)	$
E	1			1		
E'		2			3	3
T	4			4		
T'		6	5		6	6
F	8			7		

Predicative parsing table for the left-hand side grammar.

What is a predicative parsing table?

Generate string id + id from the left-most variable E':

$$E \overset{1}{\Rightarrow} TE' \overset{4}{\Rightarrow} FT'E' \overset{8}{\Rightarrow} \mathrm{id}\,T'E' \overset{6}{\Rightarrow} \mathrm{id}\,E' \overset{2}{\Rightarrow} \mathrm{id} + TE' \Rightarrow \cdots \Rightarrow \mathrm{id} + \mathrm{id}$$

Example

1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \mathrm{id}$

	id	+	*	(	)	$
E	1			1		
E'		2			3	3
T	4			4		
T'		6	5		6	6
F	8			7		

Predicative parsing table for the left-hand side grammar.

What is a predicative parsing table?

Generate string id + id from the left-most variable E':

$$E \overset{1}{\Rightarrow} TE' \overset{4}{\Rightarrow} FT'E' \overset{8}{\Rightarrow} \mathrm{id}\,T'E' \overset{6}{\Rightarrow} \mathrm{id}\,E' \overset{2}{\Rightarrow} \mathrm{id} + TE' \Rightarrow \cdots \Rightarrow \mathrm{id} + \mathrm{id}$$

Example

1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \mathrm{id}$

	id	+	*	(	)	$
E	1			1		
E'		2			3	3
T	4			4		
T'		6	5		6	6
F	8			7		

Predicative parsing table for the left-hand side grammar.

What is a predicative parsing table?

Generate string id + id from the left-most variable T:

$$E \overset{1}{\Rightarrow} TE' \overset{4}{\Rightarrow} FT'E' \overset{8}{\Rightarrow} \mathrm{id}\,T'E' \overset{6}{\Rightarrow} \mathrm{id}\,E' \overset{2}{\Rightarrow} \mathrm{id} + TE' \Rightarrow \cdots \Rightarrow \mathrm{id} + \mathrm{id}$$

Example

1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \mathrm{id}$

	id	+	*	(	)	$
E	1			1		
E'		2			3	3
T	4			4		
T'		6	5		6	6
F	8			7		

Predicative parsing table for the left-hand side grammar.

What is a predicative parsing table?

Generate string $\mathrm{id} + \mathrm{id}$ from the left-most variable T:

$$E \overset{1}{\Rightarrow} TE' \overset{4}{\Rightarrow} FT'E' \overset{8}{\Rightarrow} \mathrm{id}\,T'E' \overset{6}{\Rightarrow} \mathrm{id}E' \overset{2}{\Rightarrow} \mathrm{id} + TE' \Rightarrow \cdots \Rightarrow \mathrm{id} + \mathrm{id}$$

Example

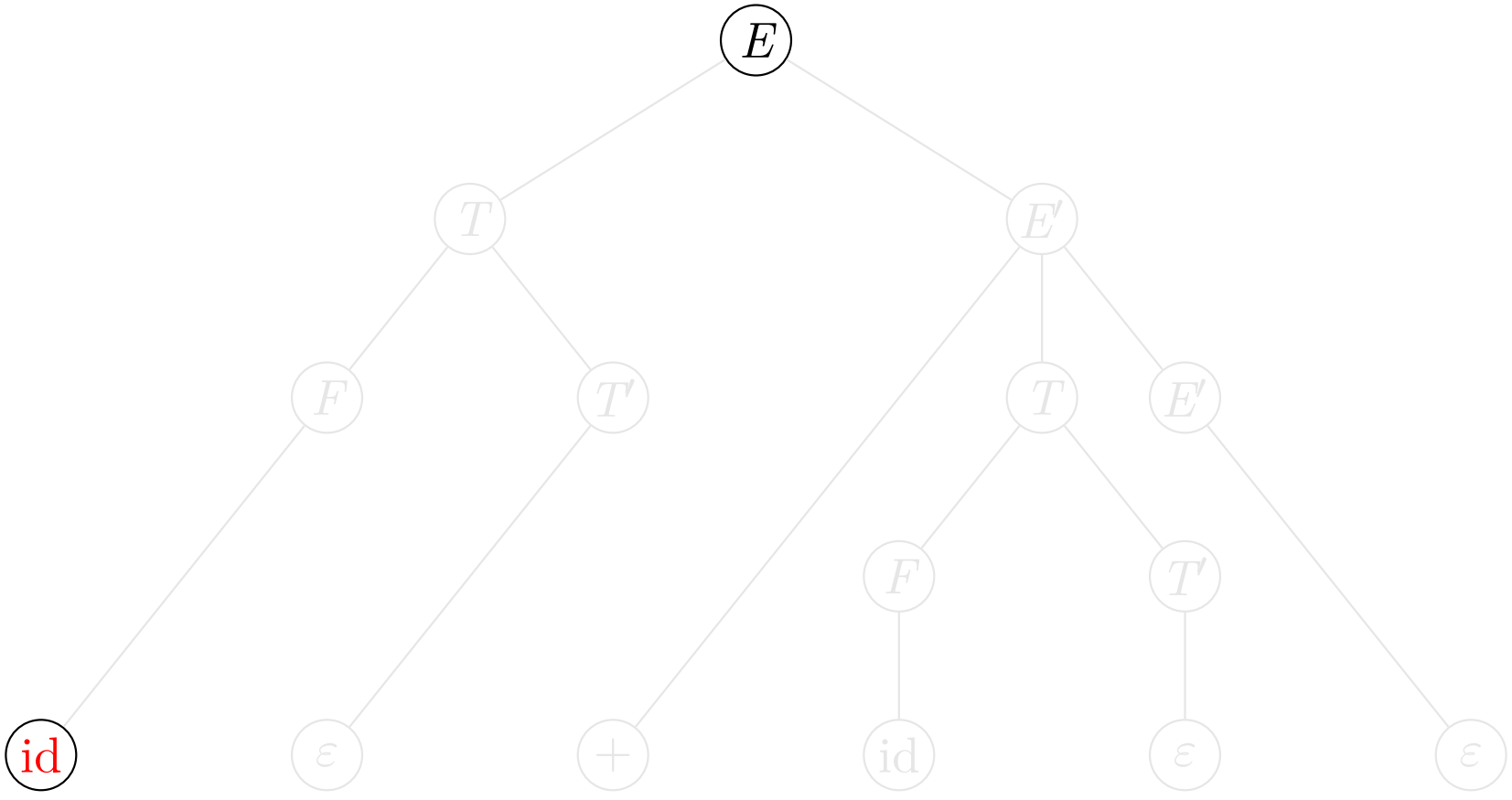
1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \mathrm{id}$

	id	+	*	(	)	$
E	1			1		
E'		2			3	3
T	4			4		
T'		6	5		6	6
F	8			7		

Predicative parsing table for the left-hand side grammar.

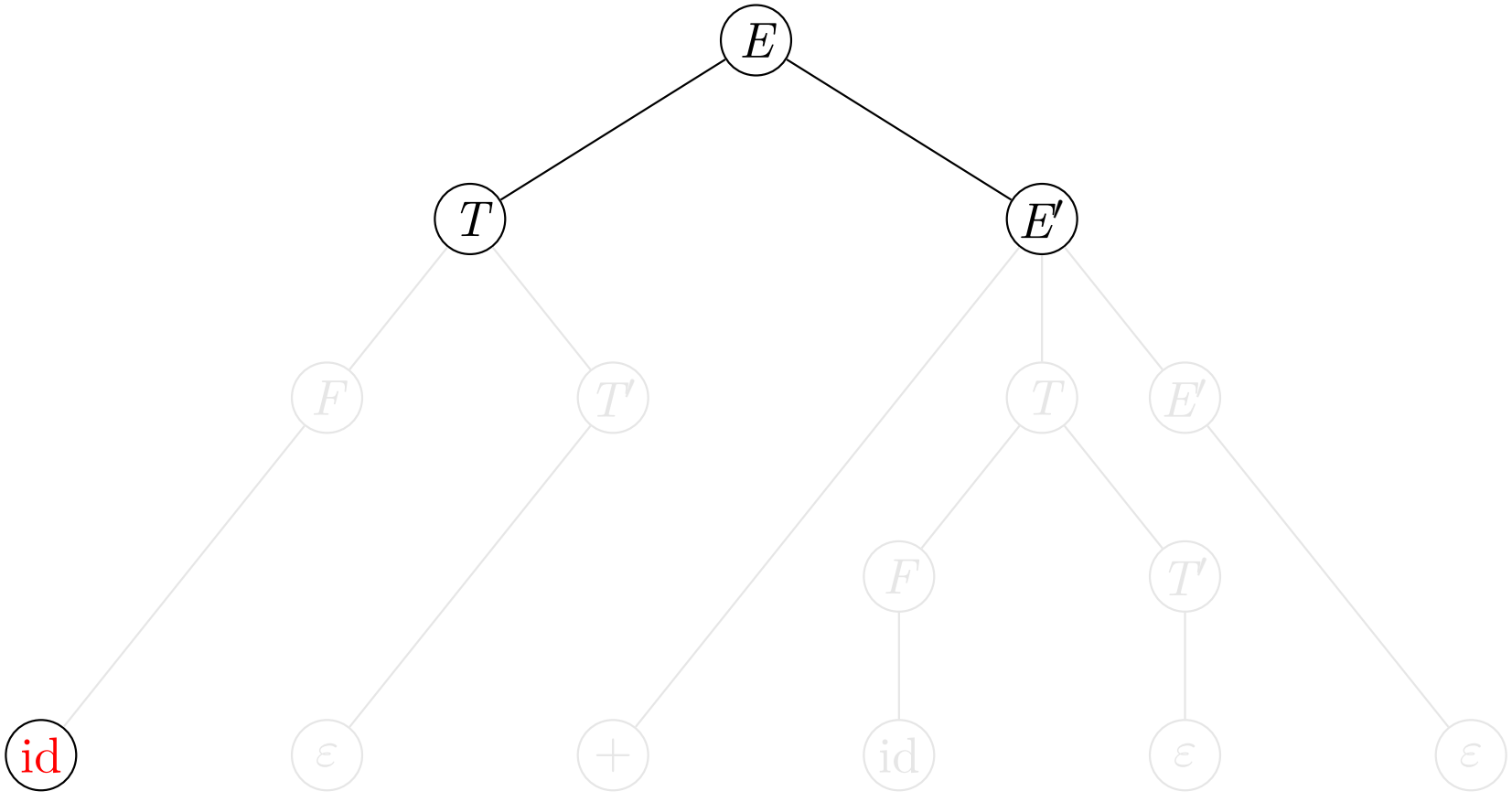
From the perspective of a parsing tree

Generate string id + id from the start variable E:



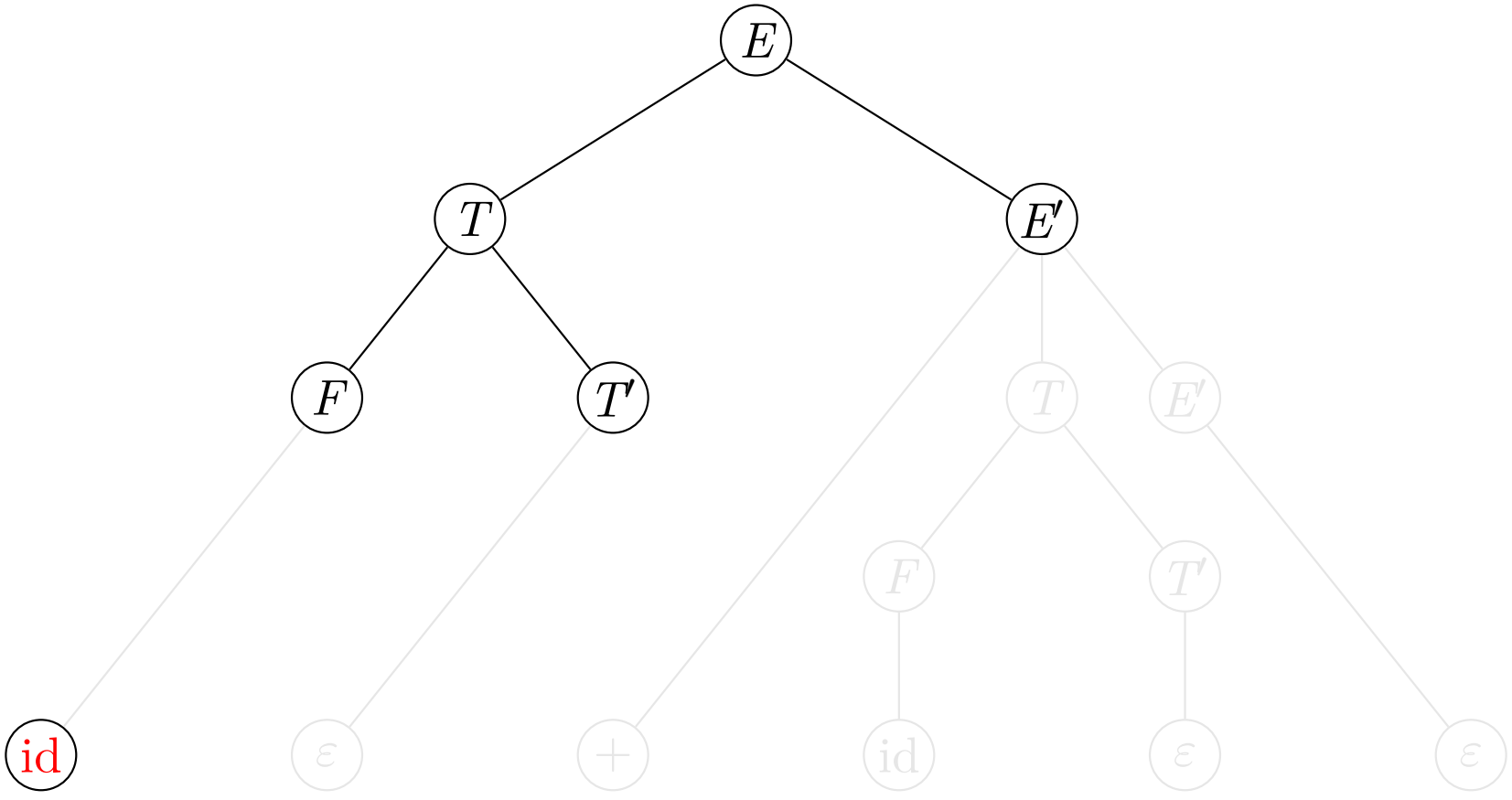
From the perspective of a parsing tree

Generate string id + id from the start variable E:



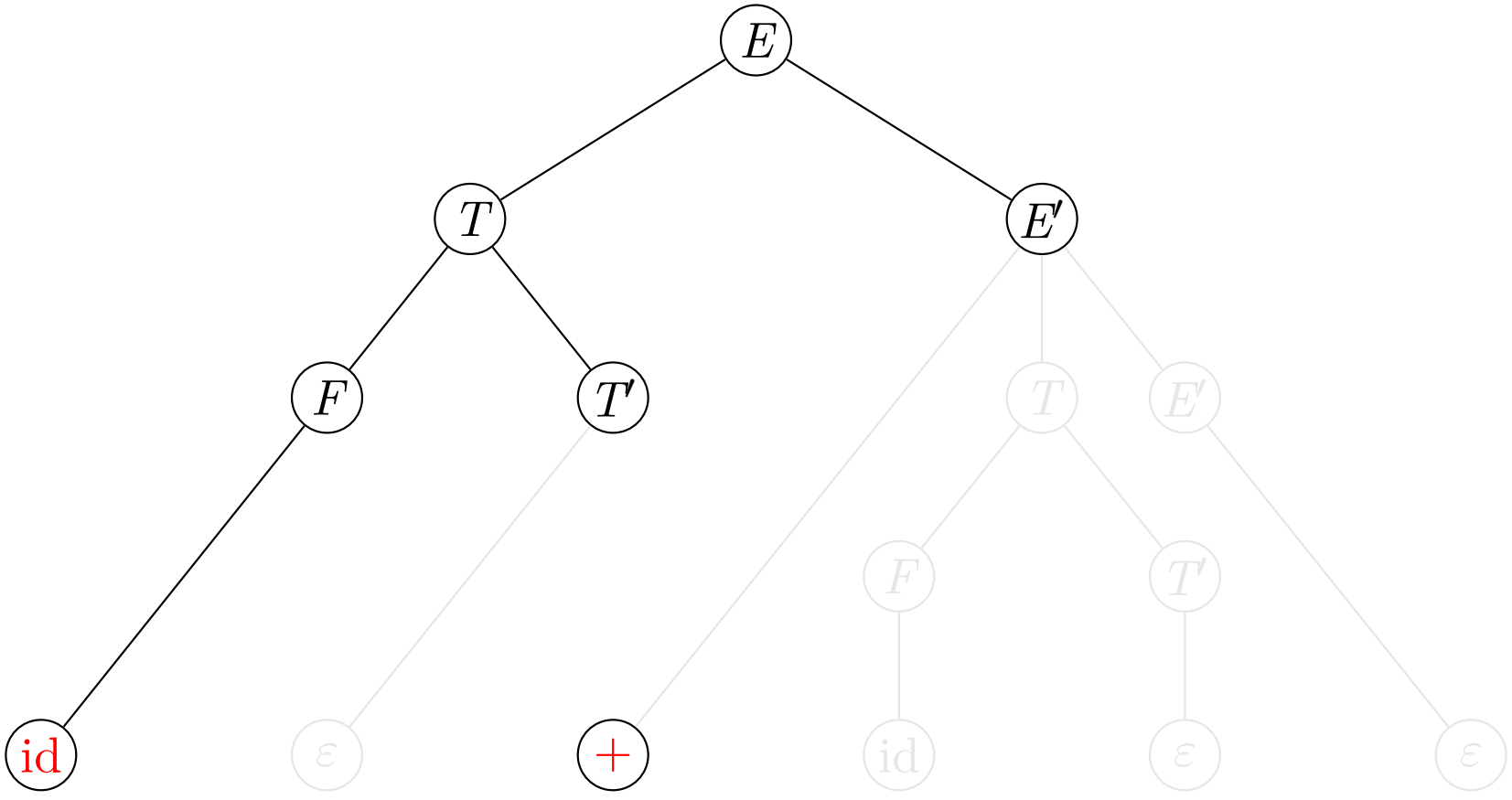
From the perspective of a parsing tree

Generate string id + id from the start variable E:



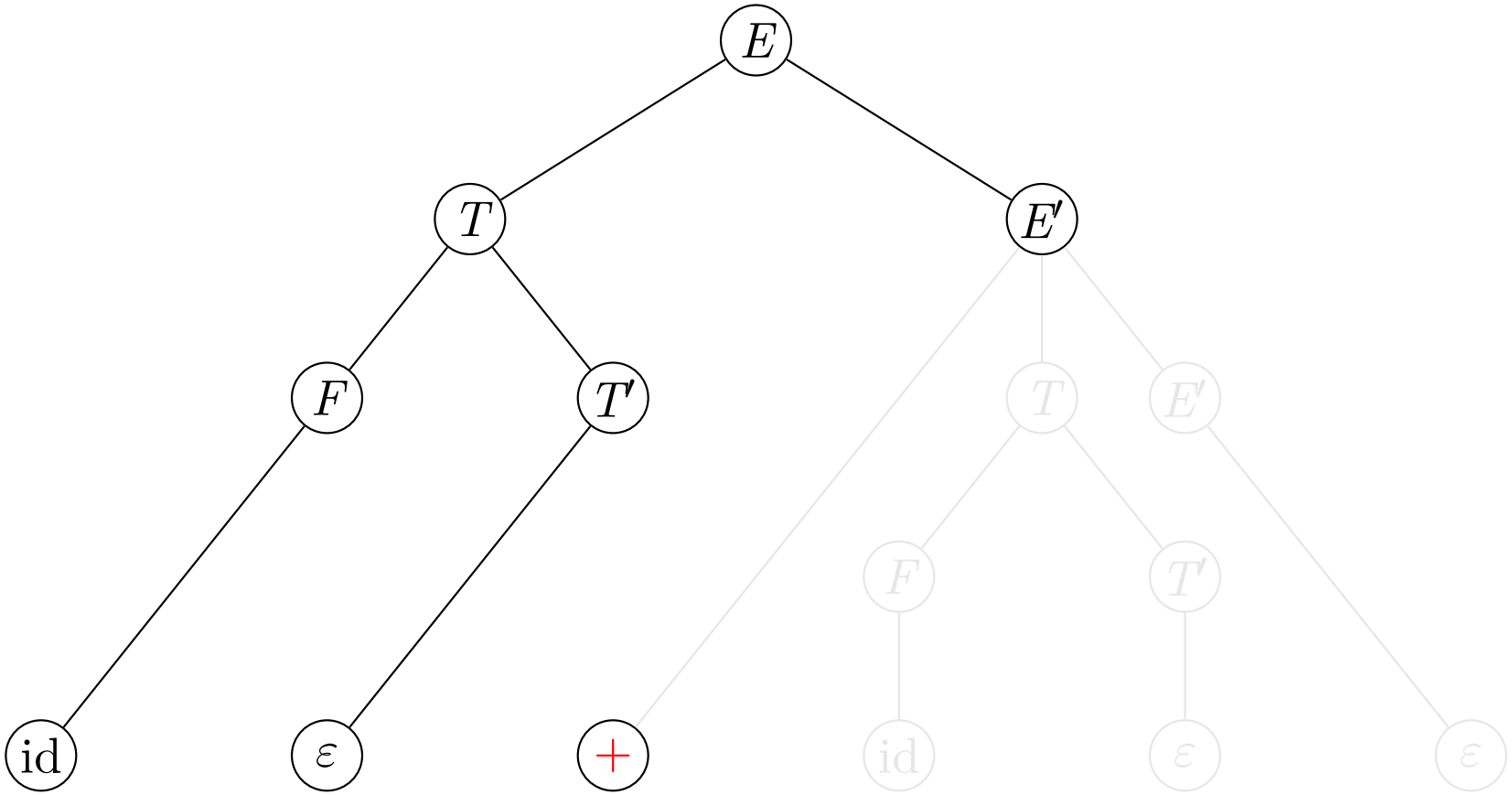
From the perspective of a parsing tree

Generate string id + id from the start variable E:



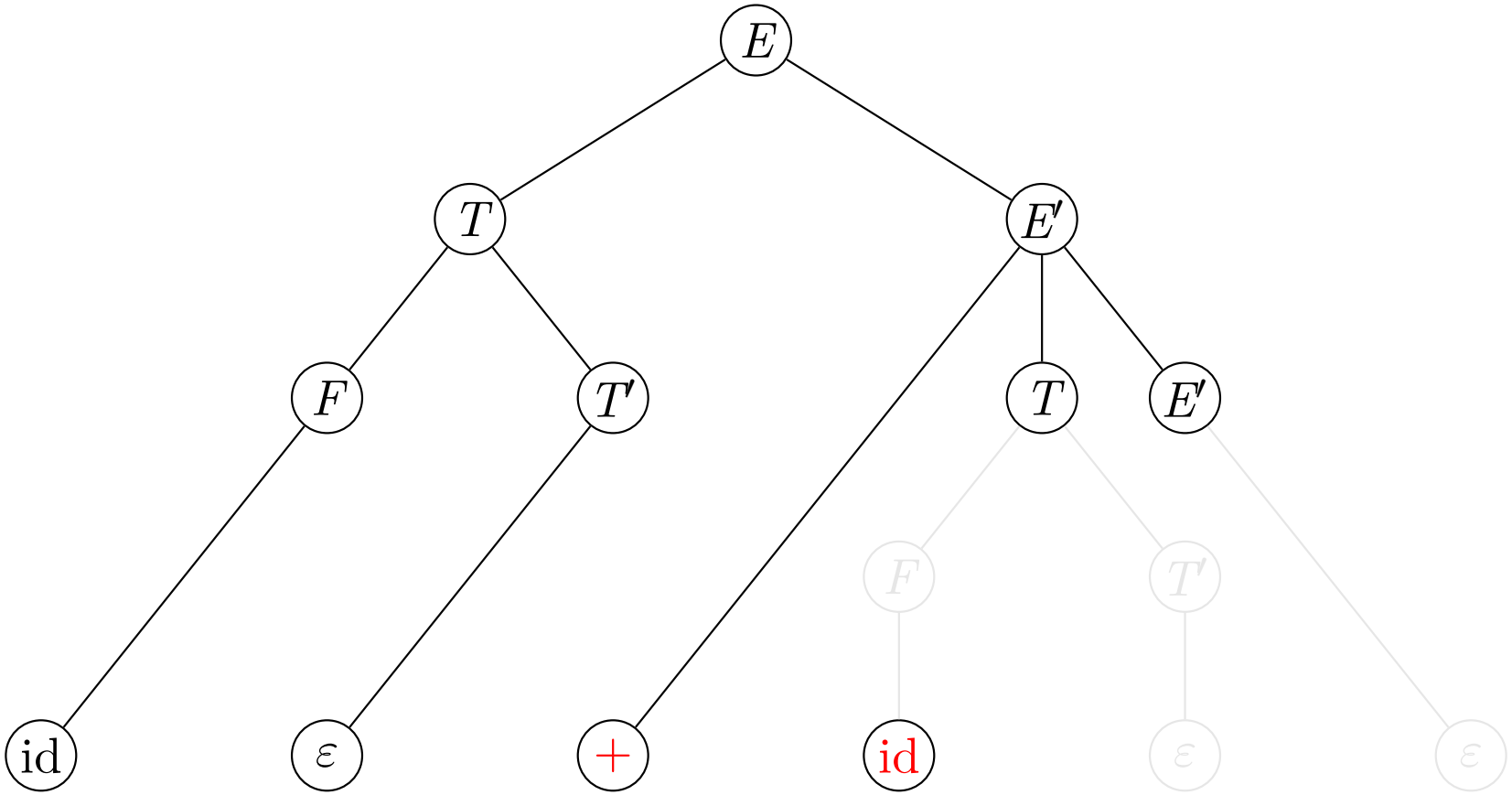
From the perspective of a parsing tree

Generate string id + id from the start variable E:



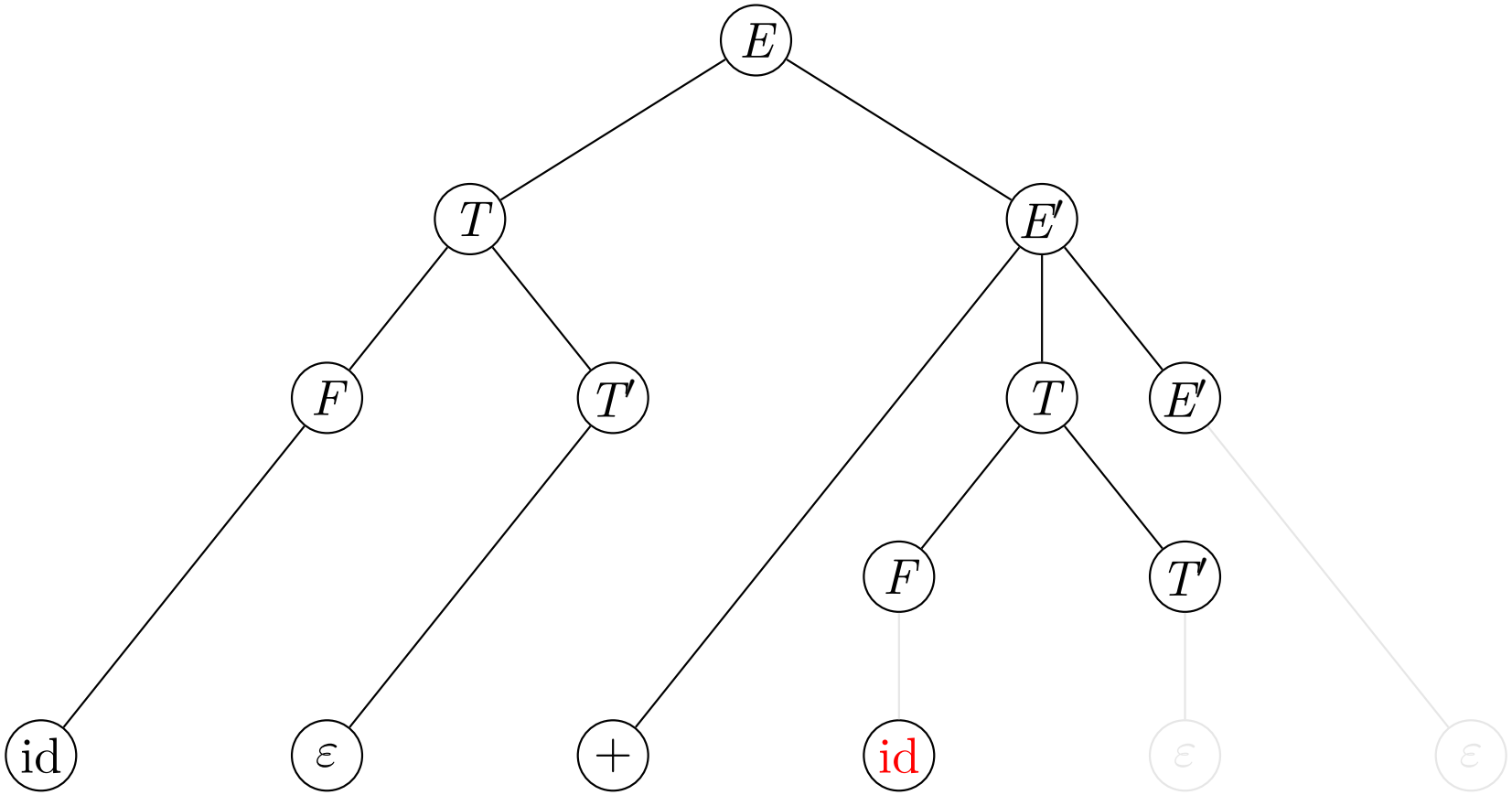
From the perspective of a parsing tree

Generate string id + id from the start variable E:



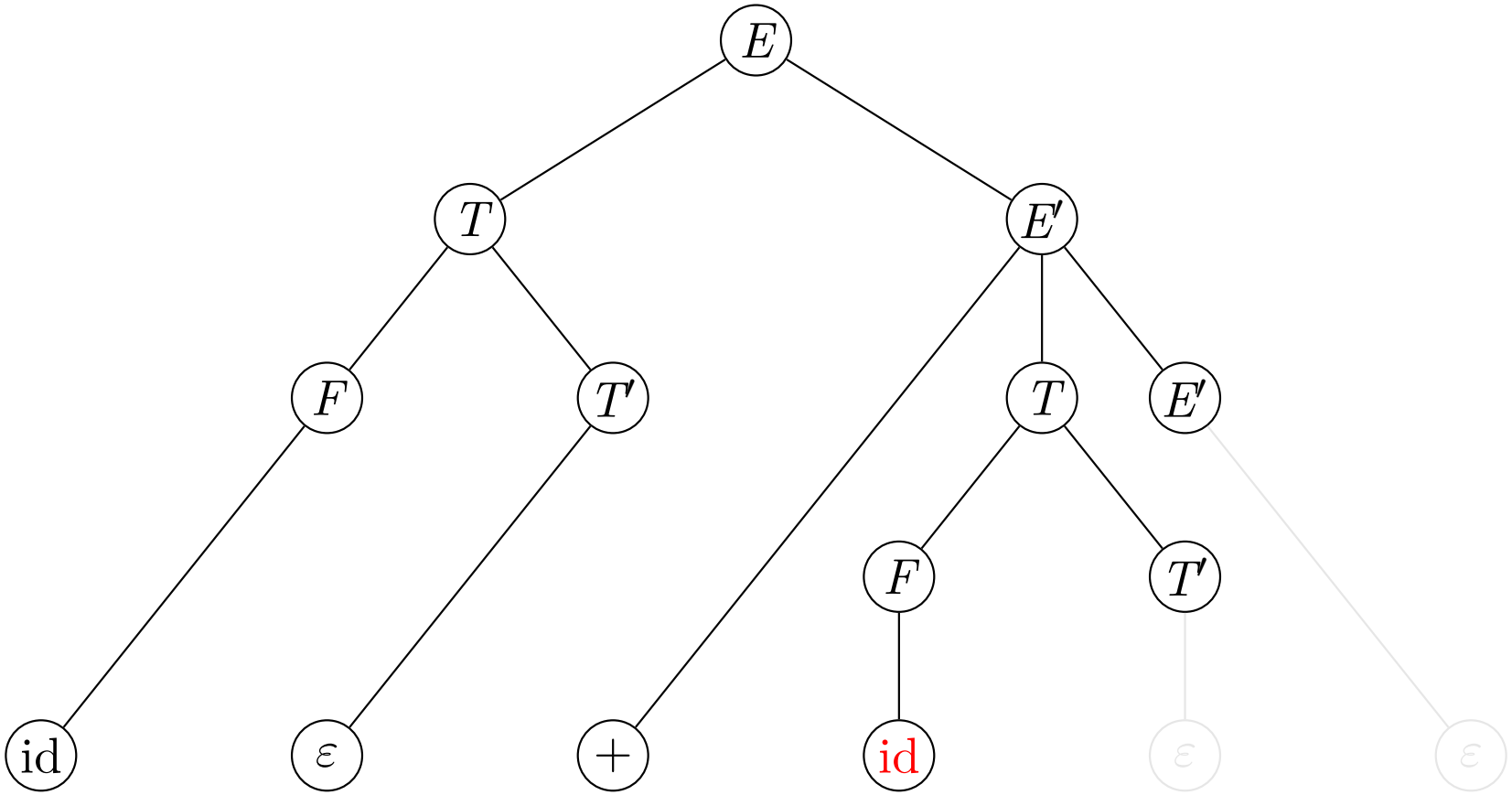
From the perspective of a parsing tree

Generate string id + id from the start variable E:



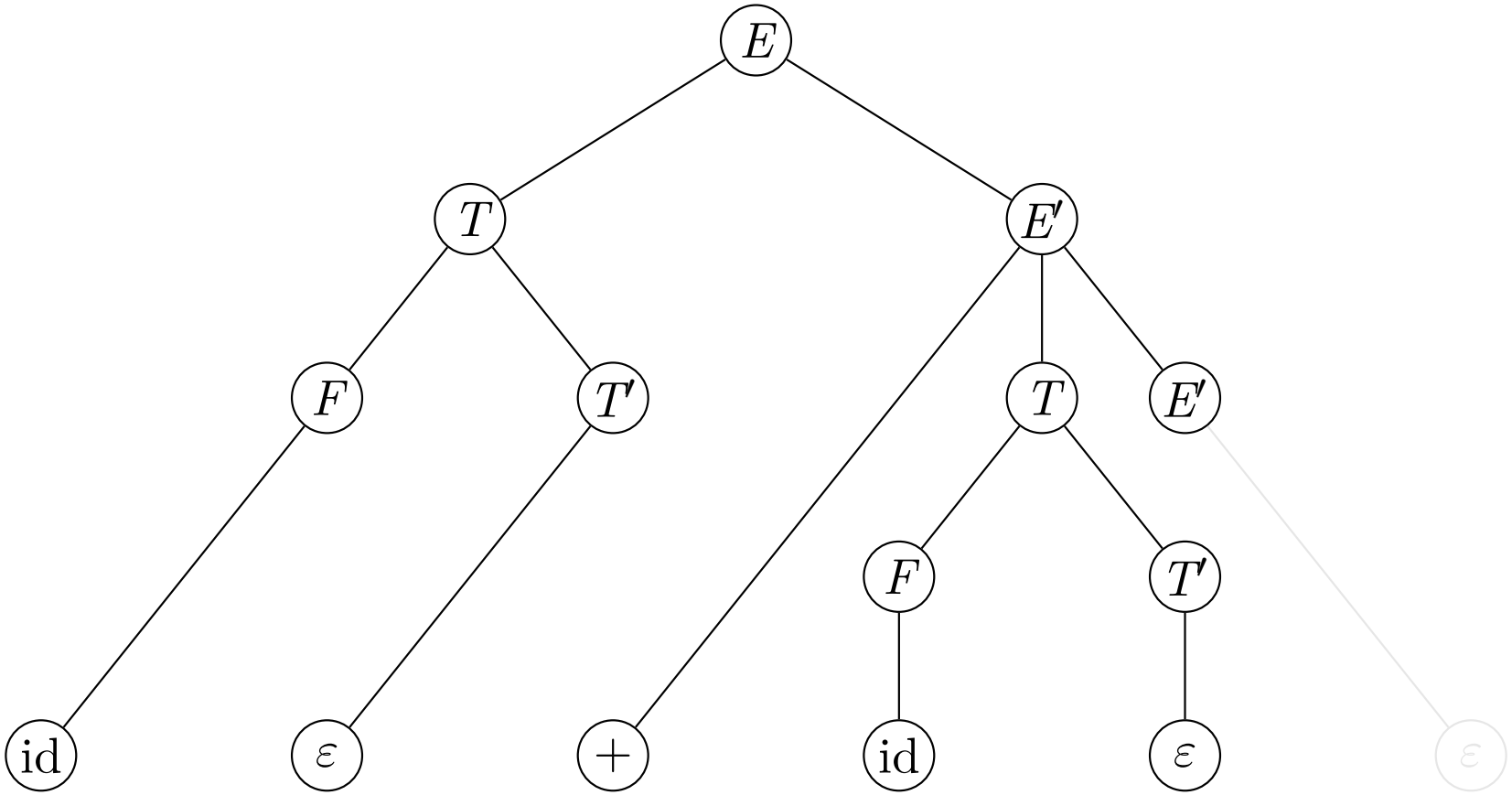
From the perspective of a parsing tree

Generate string id + id from the start variable E:



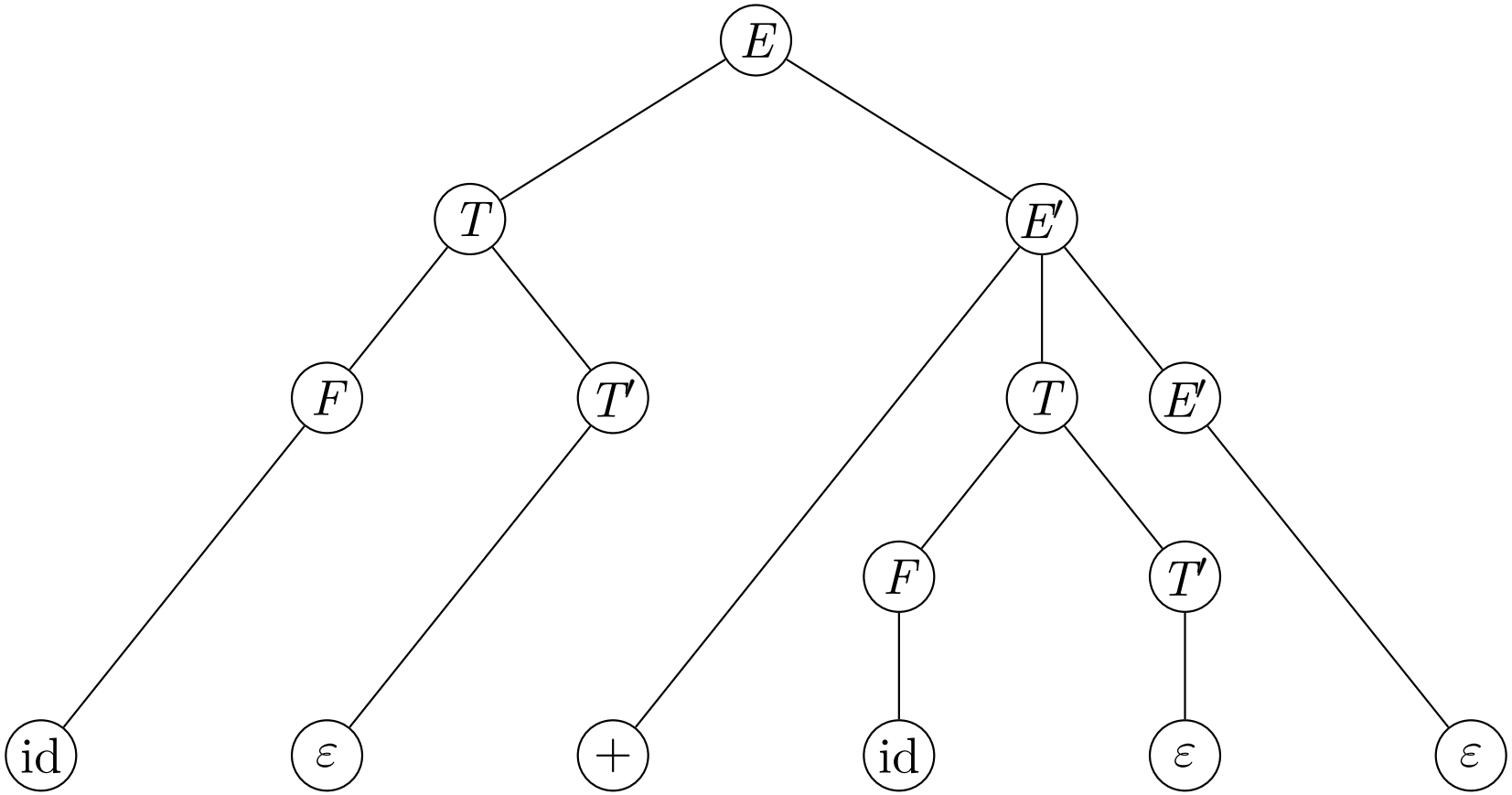
From the perspective of a parsing tree

Generate string id + id from the start variable E:



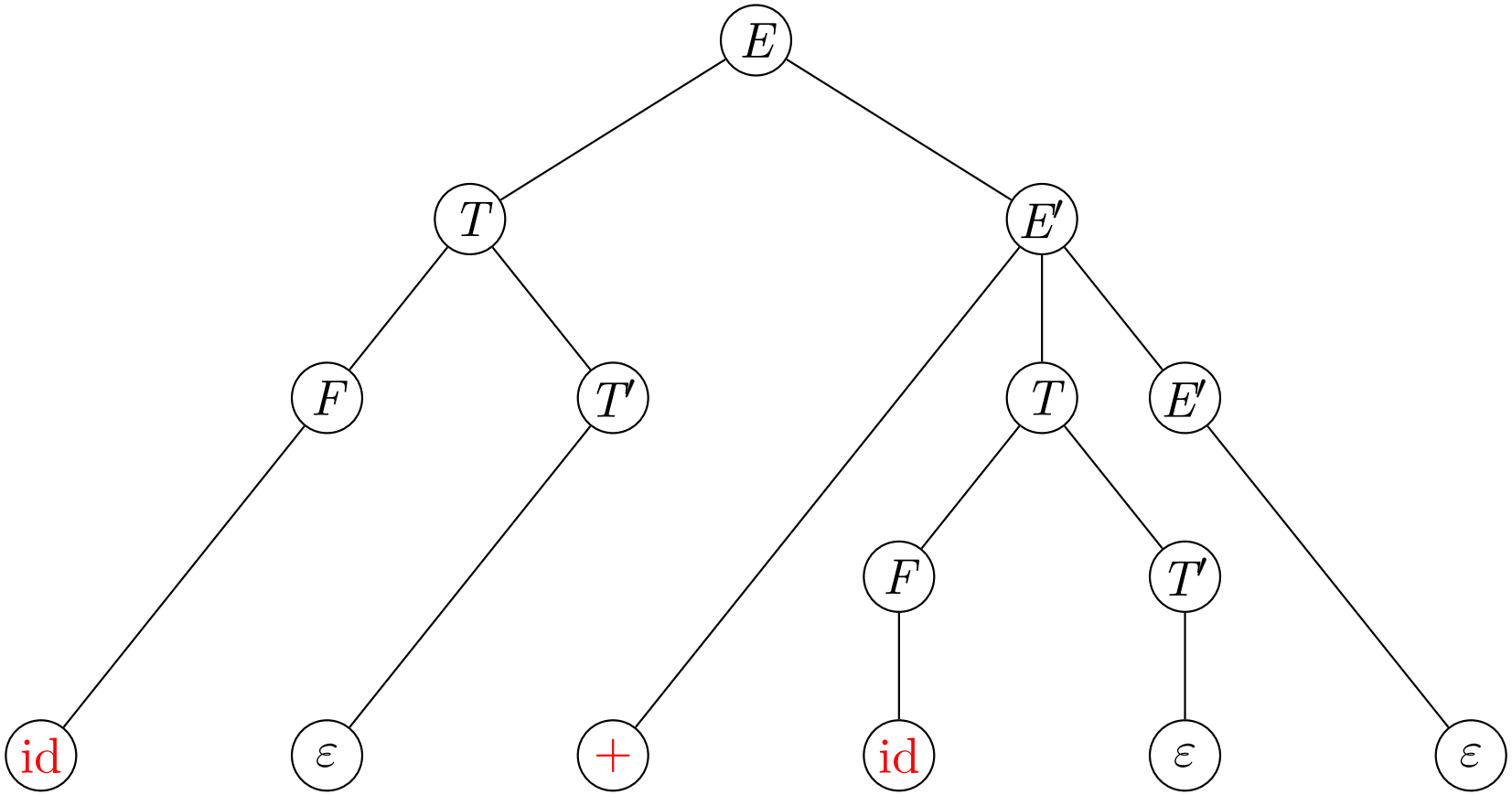
From the perspective of a parsing tree

Generate string id + id from the start variable E:



From the perspective of a parsing tree

Generate string id + id from the start variable E:



Tutorial: construction of a predicative parsing table

There are three steps:

1. construct FIRST sets;
2. construct FOLLOW sets;
3. construct the parsing table using the sets.

Tutorial: construction of a predicative parsing table

There are three steps:

1. construct FIRST sets;
2. construct FOLLOW sets;
3. construct the parsing table using the sets.

Tutorial: construction of a predicative parsing table

There are three steps:

1. construct FIRST sets;
2. construct FOLLOW sets;
3. construct the parsing table using the sets.

Tutorial: construction of a predicative parsing table

There are three steps:

1. construct FIRST sets;
2. construct FOLLOW sets;
3. construct the parsing table using the sets.

Tutorial: construction of FIRST sets

Definition (FIRST sets)

FIRST(A) is a set of symbols (including ε) that can appear in the first position of any string derived from variable A.

- If there is a rule starting with variables on both sides, we will need to incorporate the FIRST set of the latter one into that of the former one.
- What if there is one more rule of $F \to \varepsilon$ and how would this affect FIRST(T)?

1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT' \Rightarrow T'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \text{id}$

FIRST sets	
E	(, id
E'	+, ε
T	(, id
T'	*, ε
F	(, id

UTS:QSI
CENTRE FOR QUANTUM SOFTWARE AND INFORMATION

Tutorial: construction of FIRST sets

Definition (FIRST sets)

FIRST(A) is a set of symbols (including ε) that can appear in the first position of any string derived from variable A.

- If there is a rule starting with variables on both sides, we will need to incorporate the FIRST set of the latter one into that of the former one.
- What if there is one more rule of $F \to \varepsilon$ and how would this affect FIRST(T)?

1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT' \Rightarrow T'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \text{id}$

FIRST sets	
E	(, id
E'	+, ε
T	(, id
T'	*, ε
F	(, id

UTS:QSI
CENTRE FOR QUANTUM SOFTWARE AND INFORMATION

Tutorial: construction of FIRST sets

Definition (FIRST sets)

FIRST(A) is a set of symbols (including ε) that can appear in the first position of any string derived from variable A.

a If there is a rule starting with variables on both sides, we will need to incorporate the FIRST set of the latter one into that of the former one.

b What if there is one more rule of $F \to \varepsilon$ and how would this affect FIRST(T)?

1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT' \Rightarrow T'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \mathrm{id}$

FIRST sets	
E	(, id
E'	$+$, ε
T	FIRST(F)
T'	$*$, ε
F	(, id

UTS:QSI
CENTRE FOR QUANTUM SOFTWARE AND INFORMATION

Tutorial: construction of FIRST sets

Definition (FIRST sets)

FIRST(A) is a set of symbols (including ε) that can appear in the first position of any string derived from variable A.

a If there is a rule starting with variables on both sides, we will need to incorporate the FIRST set of the latter one into that of the former one.

b What if there is one more rule of $F \to \varepsilon$ and how would this affect FIRST(T)?

1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT' \Rightarrow T'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \mathrm{id}$

FIRST sets	
E	(, id
E'	+, ε
T	(, id
T'	*, ε
F	(, id

UTS:QSI
CENTRE FOR QUANTUM SOFTWARE AND INFORMATION

Tutorial: construction of FIRST sets

Definition (FIRST sets)

FIRST(A) is a set of symbols (including ε) that can appear in the first position of any string derived from variable A.

a If there is a rule starting with variables on both sides, we will need to incorporate the FIRST set of the latter one into that of the former one.

b What if there is one more rule of $F \to \varepsilon$ and how would this affect FIRST(T)?

1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT' \Rightarrow T'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \mathrm{id}$

FIRST sets

E	FIRST(T)
E'	$+, \varepsilon$
T	(, id
T'	$*, \varepsilon$
F	(, id

Tutorial: construction of FIRST sets

Definition (FIRST sets)

FIRST(A) is a set of symbols (including ε) that can appear in the first position of any string derived from variable A.

a If there is a rule starting with variables on both sides, we will need to incorporate the FIRST set of the latter one into that of the former one.

b What if there is one more rule of $F \to \varepsilon$ and how would this affect FIRST(T)?

1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT' \Rightarrow T'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \mathrm{id}$

FIRST sets	
E	(, id
E'	$+, \varepsilon$
T	(, id
T'	$*, \varepsilon$
F	(, id

UTS:QSI
CENTRE FOR QUANTUM SOFTWARE AND INFORMATION

Tutorial: construction of FIRST sets

Definition (FIRST sets)

FIRST(A) is a set of symbols (including ε) that can appear in the first position of any string derived from variable A.

a. If there is a rule starting with variables on both sides, we will need to incorporate the FIRST set of the latter one into that of the former one.

b. What if there is one more rule of $F \to \varepsilon$ and how would this affect FIRST(T)?

1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT' \Rightarrow T'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \mathrm{id}$

FIRST sets	
E	(, id
E'	$+$, ε
T	(, id
T'	$*$, ε
F	(, id

Tutorial: construction of FIRST sets

Definition (FIRST sets)

FIRST(A) is a set of symbols (including ε) that can appear in the first position of any string derived from variable A.

a If there is a rule starting with variables on both sides, we will need to incorporate the FIRST set of the latter one into that of the former one.

b What if there is one more rule of $F \to \varepsilon$ and how would this affect FIRST(T)?

1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT' \Rightarrow T'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \mathrm{id}$

FIRST sets	
E	(, id
E'	+, ε
T	(, id
T'	*, ε
F	(, id

UTS:QSI
CENTRE FOR QUANTUM SOFTWARE AND INFORMATION

Tutorial: construction of FIRST sets

Definition (FIRST sets)

FIRST(A) is a set of symbols (including ε) that can appear in the first position of any string derived from variable A.

a If there is a rule starting with variables on both sides, we will need to incorporate the FIRST set of the latter one into that of the former one.

b What if there is one more rule of $F \to \varepsilon$ and how would this affect FIRST(T)?

1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT' \Rightarrow T'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \mathrm{id}$

FIRST sets	
E	(, id
E'	+, ε
T	(, id
T'	*, ε
F	(, id

Tutorial: construction of FIRST sets

Definition (FIRST sets)

FIRST(A) is a set of symbols (including ε) that can appear in the first position of any string derived from variable A.

a If there is a rule starting with variables on both sides, we will need to incorporate the FIRST set of the latter one into that of the former one.

b What if there is one more rule of $F \to \varepsilon$ and how would this affect FIRST(T)?

1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT' \Rightarrow T'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \text{id}$

FIRST sets	
E	(, id
E'	+, ε
T	(, id
T'	*, ε
F	(, id

Tutorial: construction of FIRST sets

Definition (FIRST sets)

FIRST(A) is a set of symbols (including ε) that can appear in the first position of any string derived from variable A.

a If there is a rule starting with variables on both sides, we will need to incorporate the FIRST set of the latter one into that of the former one.

b What if there is one more rule of $F \to \varepsilon$ and how would this affect FIRST(T)?

1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT' \Rightarrow T'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \text{id}$

FIRST sets	
E	(, id
E'	+, ε
T	(, id
T'	*, ε
F	(, id

UTS:QSI
CENTRE FOR QUANTUM SOFTWARE AND INFORMATION

Tutorial: construction of FIRST sets

Definition (FIRST sets)

FIRST(A) is a set of symbols (including ε) that can appear in the first position of any string derived from variable A.

a If there is a rule starting with variables on both sides, we will need to incorporate the FIRST set of the latter one into that of the former one.

b What if there is one more rule of $F \to \varepsilon$ and how would this affect FIRST(T)?

1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT' \Rightarrow T'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \text{id}$

	FIRST sets
E	(, id
E'	$+$, ε
T	(, id, FIRST(T')
T'	$*$, ε
F	(, id

UTS:QSI
CENTRE FOR QUANTUM SOFTWARE AND INFORMATION

Tutorial: construction of FOLLOW sets

Definition (FOLLOW sets)

FOLLOW(A) is a set of symbols (not including ε) that can appear immediately after variable A in any derivation of the grammar.

1. We always incorporate \$ into the FOLLOW set of the start variable.
2. If there is a rule ending with variables on both sides, we will need to incorporate the FOLLOW set of the former one into that of the latter one.
3. If there is a rule involving two consecutive variables, we will need to incorporate the FIRST set of the latter one into the FOLLOW set of the former one.

1. $E \to TE'$
2. $E' \to +TE' \Rightarrow +T$
3. $E' \to \varepsilon$
4. $T \to FT' \Rightarrow F$
5. $T' \to *FT' \Rightarrow *F$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \mathrm{id}$

FIRST sets	
E	(, id
E'	+, ε
T	(, id
T'	*, ε
F	(, id

FOLLOW sets	
E	\$,)
E'	\$,)
T	+, \$,)
T'	+, \$,)
F	*, +, \$,)

UTS:QSI
CENTRE FOR QUANTUM SOFTWARE AND INFORMATION

Tutorial: construction of FOLLOW sets

Definition (FOLLOW sets)

FOLLOW(A) is a set of symbols (not including ε) that can appear immediately after variable A in any derivation of the grammar.

a. We always incorporate $ into the FOLLOW set of the start variable.

b. If there is a rule ending with variables on both sides, we will need to incorporate the FOLLOW set of the former one into that of the latter one.

c. If there is a rule involving two consecutive variables, we will need to incorporate the FIRST set of the latter one into the FOLLOW set of the former one. Caution if ε is in the set!

1. $E \to TE'$
2. $E' \to +TE' \Rightarrow +T$
3. $E' \to \varepsilon$
4. $T \to FT' \Rightarrow F$
5. $T' \to *FT' \Rightarrow *F$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \mathrm{id}$

FIRST sets	
E	(, id
E'	+, ε
T	(, id
T'	*, ε
F	(, id

FOLLOW sets	
E	$,)
E'	$,)
T	+, $,)
T'	+, $,)
F	*, +, $,)

UTS:QSI
CENTRE FOR QUANTUM SOFTWARE AND INFORMATION

Tutorial: construction of FOLLOW sets

Definition (FOLLOW sets)

FOLLOW(A) is a set of symbols (not including ε) that can appear immediately after variable A in any derivation of the grammar.

a. We always incorporate \$ into the FOLLOW set of the start variable.

b. If there is a rule ending with variables on both sides, we will need to incorporate the FOLLOW set of the former one into that of the latter one.

c. If there is a rule involving two consecutive variables, we will need to incorporate the FIRST set of the latter one into the FOLLOW set of the former one. Caution if ε is in the set.

1. $E \to TE'$
2. $E' \to +TE' \Rightarrow +T$
3. $E' \to \varepsilon$
4. $T \to FT' \Rightarrow F$
5. $T' \to *FT' \Rightarrow *F$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \text{id}$

FIRST sets	
E	(, id
E'	+, ε
T	(, id
T'	*, ε
F	(, id

FOLLOW sets	
E	\$,)
E'	\$,)
T	+, \$,)
T'	+, \$,)
F	*, +, \$,)

UTS:QSI
CENTRE FOR QUANTUM SOFTWARE AND INFORMATION

Tutorial: construction of FOLLOW sets

Definition (FOLLOW sets)

FOLLOW(A) is a set of symbols (not including ε) that can appear immediately after variable A in any derivation of the grammar.

a. We always incorporate \$ into the FOLLOW set of the start variable.

b. If there is a rule ending with variables on both sides, we will need to incorporate the FOLLOW set of the former one into that of the latter one.

c. If there is a rule involving two consecutive variables, we will need to incorporate the FIRST set of the latter one into the FOLLOW set of the former one. Caution if ε is in the set!

1. $E \to TE'$
2. $E' \to +TE' \Rightarrow +T$
3. $E' \to \varepsilon$
4. $T \to FT' \Rightarrow F$
5. $T' \to *FT' \Rightarrow *F$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \mathrm{id}$

	FIRST sets
E	$($, id
E'	$+$, ε
T	$($, id
T'	$*$, ε
F	$($, id

	FOLLOW sets
E	\$, $)$
E'	FOLLOW(E)
T	$+$, \$, $)$
T'	$+$, \$, $)$
F	$*$, $+$, \$, $)$

UTS:QSI
CENTRE FOR QUANTUM SOFTWARE AND INFORMATION

Tutorial: construction of FOLLOW sets

Definition (FOLLOW sets)

FOLLOW(A) is a set of symbols (not including ε) that can appear immediately after variable A in any derivation of the grammar.

a. We always incorporate \$ into the FOLLOW set of the start variable.

b. If there is a rule ending with variables on both sides, we will need to incorporate the FOLLOW set of the former one into that of the latter one.

c. If there is a rule involving two consecutive variables, we will need to incorporate the FIRST set of the latter one into the FOLLOW set of the former one. Caution if ε is in the set!

1. $E \to TE'$
2. $E' \to +TE' \Rightarrow +T$
3. $E' \to \varepsilon$
4. $T \to FT' \Rightarrow F$
5. $T' \to *FT' \Rightarrow *F$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \mathrm{id}$

FIRST sets	
E	$($, id
E'	$+$, ε
T	$($, id
T'	$*$, ε
F	$($, id

FOLLOW sets	
E	\$, $)$
E'	\$, $)$
T	$+$, \$, $)$
T'	$+$, \$, $)$
F	$*$, $+$, \$, $)$

UTS:QSI
CENTRE FOR QUANTUM SOFTWARE AND INFORMATION

Tutorial: construction of FOLLOW sets

Definition (FOLLOW sets)

FOLLOW(A) is a set of symbols (not including ε) that can appear immediately after variable A in any derivation of the grammar.

a. We always incorporate \$ into the FOLLOW set of the start variable.

b. If there is a rule ending with variables on both sides, we will need to incorporate the FOLLOW set of the former one into that of the latter one.

c. If there is a rule involving two consecutive variables, we will need to incorporate the FIRST set of the latter one into the FOLLOW set of the former one. Caution if ε is in the set!

1. $E \to TE'$
2. $E' \to +TE' \Rightarrow +T$
3. $E' \to \varepsilon$
4. $T \to FT' \Rightarrow F$
5. $T' \to *FT' \Rightarrow *F$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \text{id}$

FIRST sets	
E	(, id
E'	+, ε
T	(, id
T'	*, ε
F	(, id

FOLLOW sets	
E	\$,)
E'	\$,)
T	+, \$,)
T'	+, \$,)
F	*, +, \$,)

UTS:QSI
CENTRE FOR QUANTUM SOFTWARE AND INFORMATION

Tutorial: construction of FOLLOW sets

Definition (FOLLOW sets)

FOLLOW(A) is a set of symbols (not including ε) that can appear immediately after variable A in any derivation of the grammar.

a. We always incorporate \$ into the FOLLOW set of the start variable.

b. If there is a rule ending with variables on both sides, we will need to incorporate the FOLLOW set of the former one into that of the latter one.

c. If there is a rule involving two consecutive variables, we will need to incorporate the FIRST set of the latter one into the FOLLOW set of the former one. Caution if ε is in the set!

1. $E \to TE'$
2. $E' \to +TE' \Rightarrow +T$
3. $E' \to \varepsilon$
4. $T \to FT' \Rightarrow F$
5. $T' \to *FT' \Rightarrow *F$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \text{id}$

FIRST sets	
E	(, id
E'	+, ε
T	(, id
T'	*, ε
F	(, id

FOLLOW sets	
E	\$,)
E'	\$,)
T	+, \$,)
T'	+, \$,)
F	*, +, \$,)

UTS:QSI
CENTRE FOR QUANTUM SOFTWARE AND INFORMATION

Tutorial: construction of FOLLOW sets

Definition (FOLLOW sets)

FOLLOW(A) is a set of symbols (not including ε) that can appear immediately after variable A in any derivation of the grammar.

a. We always incorporate \$ into the FOLLOW set of the start variable.

b. If there is a rule ending with variables on both sides, we will need to incorporate the FOLLOW set of the former one into that of the latter one.

c. If there is a rule involving two consecutive variables, we will need to incorporate the FIRST set of the latter one into the FOLLOW set of the former one. Caution if ε is in the set!

1. $E \to TE'$
2. $E' \to +TE' \Rightarrow +T$
3. $E' \to \varepsilon$
4. $T \to FT' \Rightarrow F$
5. $T' \to *FT' \Rightarrow *F$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \text{id}$

FIRST sets	
E	(, id
E'	+, ε
T	(, id
T'	*, ε
F	(, id

FOLLOW sets	
E	\$,)
E'	\$,)
T	+, \$,)
T'	+, \$,)
F	*, +, \$,)

UTS:QSI
CENTRE FOR QUANTUM SOFTWARE AND INFORMATION

Tutorial: construction of FOLLOW sets

Definition (FOLLOW sets)

FOLLOW(A) is a set of symbols (not including ε) that can appear immediately after variable A in any derivation of the grammar.

a. We always incorporate \$ into the FOLLOW set of the start variable.

b. If there is a rule ending with variables on both sides, we will need to incorporate the FOLLOW set of the former one into that of the latter one.

c. If there is a rule involving two consecutive variables, we will need to incorporate the FIRST set of the latter one into the FOLLOW set of the former one. Caution if ε is in the set!

1. $E \to TE'$
2. $E' \to +TE' \Rightarrow +T$
3. $E' \to \varepsilon$
4. $T \to FT' \Rightarrow F$
5. $T' \to *FT' \Rightarrow *F$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \mathrm{id}$

FIRST sets	
E	(, id
E'	+, ε
T	(, id
T'	*, ε
F	(, id

FOLLOW sets	
E	\$,)
E'	\$,)
T	+, FOLLOW(E')
T'	+, \$,)
F	*, +, \$,)

UTS:QSI CENTRE FOR QUANTUM SOFTWARE AND INFORMATION

Tutorial: construction of FOLLOW sets

Definition (FOLLOW sets)

FOLLOW(A) is a set of symbols (not including ε) that can appear immediately after variable A in any derivation of the grammar.

a. We always incorporate \$ into the FOLLOW set of the start variable.

b. If there is a rule ending with variables on both sides, we will need to incorporate the FOLLOW set of the former one into that of the latter one.

c. If there is a rule involving two consecutive variables, we will need to incorporate the FIRST set of the latter one into the FOLLOW set of the former one. Caution if ε is in the set!

1. $E \to TE'$
2. $E' \to +TE' \Rightarrow +T$
3. $E' \to \varepsilon$
4. $T \to FT' \Rightarrow F$
5. $T' \to *FT' \Rightarrow *F$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \text{id}$

FIRST sets	
E	(, id
E'	+, ε
T	(, id
T'	*, ε
F	(, id

FOLLOW sets	
E	\$,)
E'	\$,)
T	+, \$,)
T'	+, \$,)
F	*, +, \$,)

UTS:QSI
CENTRE FOR QUANTUM SOFTWARE AND INFORMATION

Tutorial: construction of FOLLOW sets

Definition (FOLLOW sets)

FOLLOW(A) is a set of symbols (not including ε) that can appear immediately after variable A in any derivation of the grammar.

a. We always incorporate \$ into the FOLLOW set of the start variable.

b. If there is a rule ending with variables on both sides, we will need to incorporate the FOLLOW set of the former one into that of the latter one.

c. If there is a rule involving two consecutive variables, we will need to incorporate the FIRST set of the latter one into the FOLLOW set of the former one. Caution if ε is in the set!

1. $E \to TE'$
2. $E' \to +TE' \Rightarrow +T$
3. $E' \to \varepsilon$
4. $T \to FT' \Rightarrow F$
5. $T' \to *FT' \Rightarrow *F$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \text{id}$

FIRST sets

E	(, id
E'	+, ε
T	(, id
T'	*, ε
F	(, id

FOLLOW sets

E	\$,)
E'	\$,)
T	+, \$,)
T'	FOLLOW(T)
F	*, +, \$,)

UTS:QSI
CENTRE FOR QUANTUM SOFTWARE AND INFORMATION

Tutorial: construction of FOLLOW sets

Definition (FOLLOW sets)

FOLLOW(A) is a set of symbols (not including ε) that can appear immediately after variable A in any derivation of the grammar.

a. We always incorporate \$ into the FOLLOW set of the start variable.

b. If there is a rule ending with variables on both sides, we will need to incorporate the FOLLOW set of the former one into that of the latter one.

c. If there is a rule involving two consecutive variables, we will need to incorporate the FIRST set of the latter one into the FOLLOW set of the former one. Caution if ε is in the set!

1. $E \to TE'$
2. $E' \to +TE' \Rightarrow +T$
3. $E' \to \varepsilon$
4. $T \to FT' \Rightarrow F$
5. $T' \to *FT' \Rightarrow *F$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \mathrm{id}$

FIRST sets	
E	(, id
E'	+, ε
T	(, id
T'	*, ε
F	(, id

FOLLOW sets	
E	\$,)
E'	\$,)
T	+, \$,)
T'	+, \$,)
F	*, +, \$,)

UTS:QSI
CENTRE FOR QUANTUM SOFTWARE AND INFORMATION

Tutorial: construction of FOLLOW sets

Definition (FOLLOW sets)

FOLLOW(A) is a set of symbols (not including ε) that can appear immediately after variable A in any derivation of the grammar.

a. We always incorporate \$ into the FOLLOW set of the start variable.

b. If there is a rule ending with variables on both sides, we will need to incorporate the FOLLOW set of the former one into that of the latter one.

c. If there is a rule involving two consecutive variables, we will need to incorporate the FIRST set of the latter one into the FOLLOW set of the former one. Caution if ε is in the set!

1. $E \to TE'$
2. $E' \to +TE' \Rightarrow +T$
3. $E' \to \varepsilon$
4. $T \to FT' \Rightarrow F$
5. $T' \to *FT' \Rightarrow *F$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \text{id}$

FIRST sets	
E	(, id
E'	+, ε
T	(, id
T'	*, ε
F	(, id

FOLLOW sets	
E	\$,)
E'	\$,)
T	+, \$,)
T'	+, \$,)
F	*, +, \$,)

UTS:QSI
CENTRE FOR QUANTUM SOFTWARE AND INFORMATION

Tutorial: construction of FOLLOW sets

Definition (FOLLOW sets)

FOLLOW(A) is a set of symbols (not including ε) that can appear immediately after variable A in any derivation of the grammar.

a. We always incorporate \$ into the FOLLOW set of the start variable.

b. If there is a rule ending with variables on both sides, we will need to incorporate the FOLLOW set of the former one into that of the latter one.

c. If there is a rule involving two consecutive variables, we will need to incorporate the FIRST set of the latter one into the FOLLOW set of the former one. Caution if ε is in the set!

1. $E \to TE'$
2. $E' \to +TE' \Rightarrow +T$
3. $E' \to \varepsilon$
4. $T \to FT' \Rightarrow F$
5. $T' \to *FT' \Rightarrow *F$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \text{id}$

FIRST sets	
E	(, id
E'	+, ε
T	(, id
T'	*, ε
F	(, id

FOLLOW sets	
E	\$,)
E'	\$,)
T	+, \$,)
T'	+, \$,)
F	*, +, \$,)

UTS:QSI
CENTRE FOR QUANTUM SOFTWARE AND INFORMATION

Tutorial: construction of FOLLOW sets

Definition (FOLLOW sets)

FOLLOW(A) is a set of symbols (not including ε) that can appear immediately after variable A in any derivation of the grammar.

a. We always incorporate \$ into the FOLLOW set of the start variable.

b. If there is a rule ending with variables on both sides, we will need to incorporate the FOLLOW set of the former one into that of the latter one.

c. If there is a rule involving two consecutive variables, we will need to incorporate the FIRST set of the latter one into the FOLLOW set of the former one. Caution if ε is in the set!

1. $E \to TE'$
2. $E' \to +TE' \Rightarrow +T$
3. $E' \to \varepsilon$
4. $T \to FT' \Rightarrow F$
5. $T' \to *FT' \Rightarrow *F$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \text{id}$

FIRST sets	
E	(, id
E'	+, ε
T	(, id
T'	*, ε
F	(, id

FOLLOW sets	
E	\$,)
E'	\$,)
T	+, \$,)
T'	+, \$,)
F	*, +, \$,)

UTS:QSI
CENTRE FOR QUANTUM SOFTWARE AND INFORMATION

Tutorial: construction of FOLLOW sets

Definition (FOLLOW sets)

FOLLOW(A) is a set of symbols (not including ε) that can appear immediately after variable A in any derivation of the grammar.

a. We always incorporate \$ into the FOLLOW set of the start variable.

b. If there is a rule ending with variables on both sides, we will need to incorporate the FOLLOW set of the former one into that of the latter one.

c. If there is a rule involving two consecutive variables, we will need to incorporate the FIRST set of the latter one into the FOLLOW set of the former one. Caution if ε is in the set!

1. $E \to TE'$
2. $E' \to +TE' \Rightarrow +T$
3. $E' \to \varepsilon$
4. $T \to FT' \Rightarrow F$
5. $T' \to *FT' \Rightarrow *F$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \mathrm{id}$

FIRST sets	
E	(, id
E'	+, ε
T	(, id
T'	*, ε
F	(, id

	FOLLOW sets
E	\$,)
E'	\$,)
T	+, \$,)
T'	+, \$,)
F	*, FOLLOW(T, T')

UTS:QSI
CENTRE FOR QUANTUM SOFTWARE AND INFORMATION

Tutorial: construction of FOLLOW sets

Definition (FOLLOW sets)

FOLLOW(A) is a set of symbols (not including ε) that can appear immediately after variable A in any derivation of the grammar.

a. We always incorporate \$ into the FOLLOW set of the start variable.

b. If there is a rule ending with variables on both sides, we will need to incorporate the FOLLOW set of the former one into that of the latter one.

c. If there is a rule involving two consecutive variables, we will need to incorporate the FIRST set of the latter one into the FOLLOW set of the former one. Caution if ε is in the set!

1. $E \to TE'$
2. $E' \to +TE' \Rightarrow +T$
3. $E' \to \varepsilon$
4. $T \to FT' \Rightarrow F$
5. $T' \to *FT' \Rightarrow *F$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \mathrm{id}$

FIRST sets	
E	$($, id
E'	$+$, ε
T	$($, id
T'	$*$, ε
F	$($, id

FOLLOW sets	
E	\$, $)$
E'	\$, $)$
T	$+$, \$, $)$
T'	$+$, \$, $)$
F	$*$, $+$, \$, $)$

UTS:QSI
CENTRE FOR QUANTUM SOFTWARE AND INFORMATION

Tutorial: construction of FOLLOW sets

Definition (FOLLOW sets)

FOLLOW(A) is a set of symbols (not including ε) that can appear immediately after variable A in any derivation of the grammar.

a. We always incorporate \$ into the FOLLOW set of the start variable.

b. If there is a rule ending with variables on both sides, we will need to incorporate the FOLLOW set of the former one into that of the latter one.

c. If there is a rule involving two consecutive variables, we will need to incorporate the FIRST set of the latter one into the FOLLOW set of the former one. Caution if ε is in the set!

1. $E \to TE'$
2. $E' \to +TE' \Rightarrow +T$
3. $E' \to \varepsilon$
4. $T \to FT' \Rightarrow F$
5. $T' \to *FT' \Rightarrow *F$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \mathrm{id}$

FIRST sets	
E	(, id
E'	+, ε
T	(, id
T'	*, ε
F	(, id

FOLLOW sets	
E	\$,)
E'	\$,)
T	+, \$,)
T'	+, \$,)
F	*, +, \$,)

UTS:QSI CENTRE FOR QUANTUM SOFTWARE AND INFORMATION

Tutorial: construction of parsing table using the sets

	id	+	*	(	)	$
E	1			1		
E'		2			3	3
T	4			4		
T'		6	5		6	6
F	8			7		

1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \text{id}$

FIRST sets	
E	(, id
E'	+, ε
T	(, id
T'	*, ε
F	(, id

FOLLOW sets	
E	\$,)
E'	\$,)
T	+, \$,)
T'	+, \$,)
F	*, +, \$,)

Tutorial: construction of parsing table using the sets

$E \to TE'$ means E can go to the terminals in FIRST(T).

	id	+	*	(	)	$
E	1			1		
E'		2			3	3
T	4			4		
T'		6	5		6	6
F	8			7		

1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \text{id}$

FIRST sets	
E	(, id
E'	+, ε
T	(, id
T'	*, ε
F	(, id

FOLLOW sets	
E	\$,)
E'	\$,)
T	+, \$,)
T'	+, \$,)
F	*, +, \$,)

UTS:QSI
CENTRE FOR QUANTUM SOFTWARE AND INFORMATION

Tutorial: construction of parsing table using the sets

$E \to TE'$ means E can go to the terminals in FIRST(T).

	id	+	*	(	)	$
E	1			1		
E'		2			3	3
T	4			4		
T'		6	5		6	6
F	8			7		

1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \text{id}$

FIRST sets	
E	(, id
E'	+, ε
T	(, id
T'	*, ε
F	(, id

FOLLOW sets	
E	\$,)
E'	\$,)
T	+, \$,)
T'	+, \$,)
F	*, +, \$,)

UTS:QSI
CENTRE FOR QUANTUM SOFTWARE AND INFORMATION

Tutorial: construction of parsing table using the sets

$E' \to +TE'$ means E' can go to terminal $+$.

	id	$+$	$*$	(	)	$
E	1			1		
E'		2			3	3
T	4			4		
T'		6	5		6	6
F	8			7		

1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \text{id}$

FIRST sets	
E	(, id
E'	$+$, ε
T	(, id
T'	$*$, ε
F	(, id

FOLLOW sets	
E	\$,)
E'	\$,)
T	$+$, \$,)
T'	$+$, \$,)
F	$*$, $+$, \$,)

Tutorial: construction of parsing table using the sets

$E' \to \varepsilon$ means E' can go to the terminals in FOLLOW(E').

	id	+	*	(	)	$
E	1			1		
E'		2			3	3
T	4			4		
T'		6	5		6	6
F	8			7		

1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \text{id}$

FIRST sets	
E	(, id
E'	+, ε
T	(, id
T'	*, ε
F	(, id

FOLLOW sets	
E	\$,)
E'	\$,)
T	+, \$,)
T'	+, \$,)
F	*, +, \$,)

UTS:QSI
CENTRE FOR QUANTUM SOFTWARE AND INFORMATION

Tutorial: construction of parsing table using the sets

$E' \to \varepsilon$ means E' can go to the terminals in FOLLOW(E').

	id	+	*	(	)	$
E	1			1		
E'		2			3	3
T	4			4		
T'		6	5		6	6
F	8			7		

1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \text{id}$

FIRST sets	
E	(, id
E'	+, ε
T	(, id
T'	*, ε
F	(, id

FOLLOW sets	
E	\$,)
E'	\$,)
T	+, \$,)
T'	+, \$,)
F	*, +, \$,)

UTS:QSI
CENTRE FOR QUANTUM SOFTWARE AND INFORMATION

Tutorial: construction of parsing table using the sets

$T \to FT'$ means T can go to the terminals in FIRST(F).

	id	+	*	(	)	$
E	1			1		
E'		2			3	3
T	4			4		
T'		6	5		6	6
F	8			7		

1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \text{id}$

FIRST sets	
E	(, id
E'	+, ε
T	(, id
T'	*, ε
F	(, id

FOLLOW sets	
E	\$,)
E'	\$,)
T	+, \$,)
T'	+, \$,)
F	*, +, \$,)

Tutorial: construction of parsing table using the sets

$T \to FT'$ means T can go to the terminals in FIRST(F).

	id	+	*	(	)	$
E	1			1		
E'		2			3	3
T	4			4		
T'		6	5		6	6
F	8			7		

1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \text{id}$

FIRST sets	
E	(, id
E'	+, ε
T	(, id
T'	*, ε
F	(, id

FOLLOW sets	
E	\$,)
E'	\$,)
T	+, \$,)
T'	+, \$,)
F	*, +, \$,)

Tutorial: construction of parsing table using the sets

$T' \to *FT'$ means T' can go to terminal $*$.

	id	+	*	(	)	$
E	1			1		
E'		2			3	3
T	4			4		
T'		6	5		6	6
F	8			7		

1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \text{id}$

FIRST sets	
E	(, id
E'	+, ε
T	(, id
T'	*, ε
F	(, id

FOLLOW sets	
E	\$,)
E'	\$,)
T	+, \$,)
T'	+, \$,)
F	*, +, \$,)

UTS:QSI
CENTRE FOR QUANTUM SOFTWARE AND INFORMATION

Tutorial: construction of parsing table using the sets

$T' \to \varepsilon$ means T' can go to the terminals in FOLLOW(T').

	id	+	*	(	)	$
E	1			1		
E'		2			3	3
T	4			4		
T'		6	5		6	6
F	8			7		

1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \text{id}$

FIRST sets	
E	(, id
E'	+, ε
T	(, id
T'	*, ε
F	(, id

FOLLOW sets	
E	\$,)
E'	\$,)
T	+, \$,)
T'	+, \$,)
F	*, +, \$,)

UTS:QSI
CENTRE FOR QUANTUM SOFTWARE AND INFORMATION

Tutorial: construction of parsing table using the sets

$T' \to \varepsilon$ means T' can go to the terminals in FOLLOW(T').

	id	+	*	(	)	$
E	1			1		
E'		2			3	3
T	4			4		
T'		6	5		6	6
F	8			7		

1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \text{id}$

FIRST sets	
E	(, id
E'	+, ε
T	(, id
T'	*, ε
F	(, id

FOLLOW sets	
E	\$,)
E'	\$,)
T	+, \$,)
T'	+, \$,)
F	*, +, \$,)

UTS:QSI
CENTRE FOR QUANTUM SOFTWARE AND INFORMATION

Tutorial: construction of parsing table using the sets

$F \to (E)$ means F can go to terminal (.

	id	+	*	(	)	$
E	1			1		
E'		2			3	3
T	4			4		
T'		6	5		6	6
F	8			7		

1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \text{id}$

FIRST sets	
E	(, id
E'	+, ε
T	(, id
T'	*, ε
F	(, id

FOLLOW sets	
E	\$,)
E'	\$,)
T	+, \$,)
T'	+, \$,)
F	*, +, \$,)

Tutorial: construction of parsing table using the sets

$F \to \text{id}$ means F can go to terminal id.

	id	+	*	(	)	$
E	1			1		
E'		2			3	3
T	4			4		
T'		6	5		6	6
F	8			7		

1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \text{id}$

FIRST sets	
E	(, id
E'	+, ε
T	(, id
T'	*, ε
F	(, id

FOLLOW sets	
E	\$,)
E'	\$,)
T	+, \$,)
T'	+, \$,)
F	*, +, \$,)

UTS:QSI
CENTRE FOR QUANTUM SOFTWARE AND INFORMATION

Tutorial: construction of parsing table using the sets

	id	+	*	(	)	$
E	1			1		
E'		2			3	3
T	4			4		
T'		6	5		6	6
F	8			7		

1. $E \to TE'$
2. $E' \to +TE'$
3. $E' \to \varepsilon$
4. $T \to FT'$
5. $T' \to *FT'$
6. $T' \to \varepsilon$
7. $F \to (E)$
8. $F \to \text{id}$

FIRST sets	
E	(, id
E'	+, ε
T	(, id
T'	*, ε
F	(, id

FOLLOW sets	
E	\$,)
E'	\$,)
T	+, \$,)
T'	+, \$,)
F	*, +, \$,)