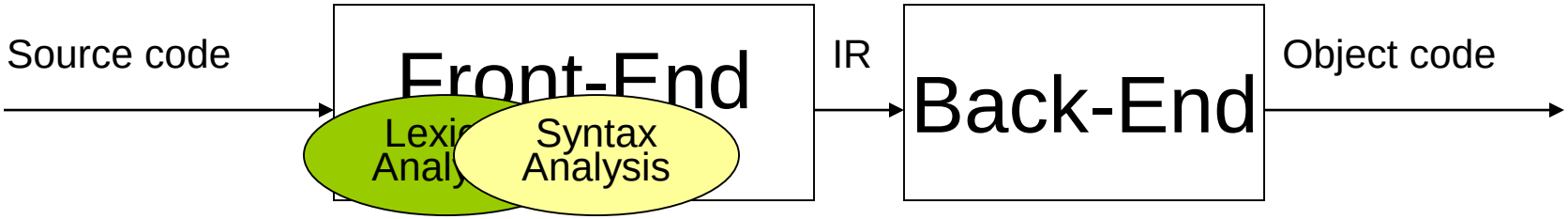
Compiler Design

Lecture 6:
Syntax Analysis
Bottom-Up Parsing (part III)

Dr. Momtazi
momtazi@aut.ac.ir

based on the slides of the course book

Parsing (Syntax Analysis)



- Syntax Analysis:
  - Derivation and parse trees ✓
  - Top-down parsing ✓
  - **Bottom-up parsing**

Outline

- Introduction
- Shift-reduce Parsing
- **LR Parsing**
  - LR(0)
  - **LR(1)**
  - SLR
  - LALR
- Ambiguity
- Error-handling

A Powerful Parser: LR(1)

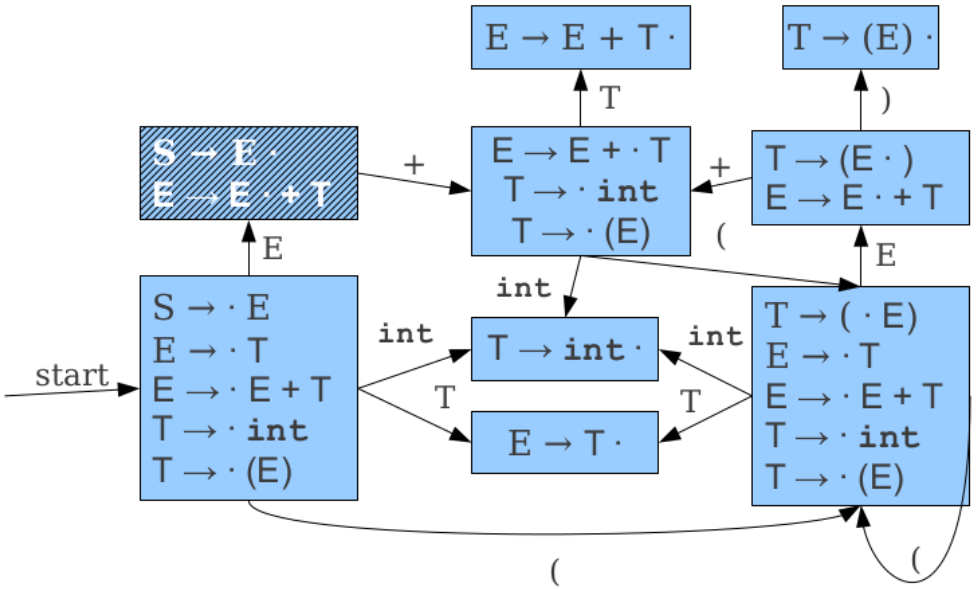
- Bottom-up predictive parsing with
  - L: Left-to-right scan
  - R: Rightmost derivation
  - (1): One token lookahead
- Substantially more powerful than the other methods we've covered so far
- Tries to more intelligently find handles by using a lookahead token at each step.

LR(1) Parsing: The Intuition

$S \rightarrow E$
$E \rightarrow T$
$E \rightarrow E + T$
$T \rightarrow \texttt{int}$
$T \rightarrow (E)$

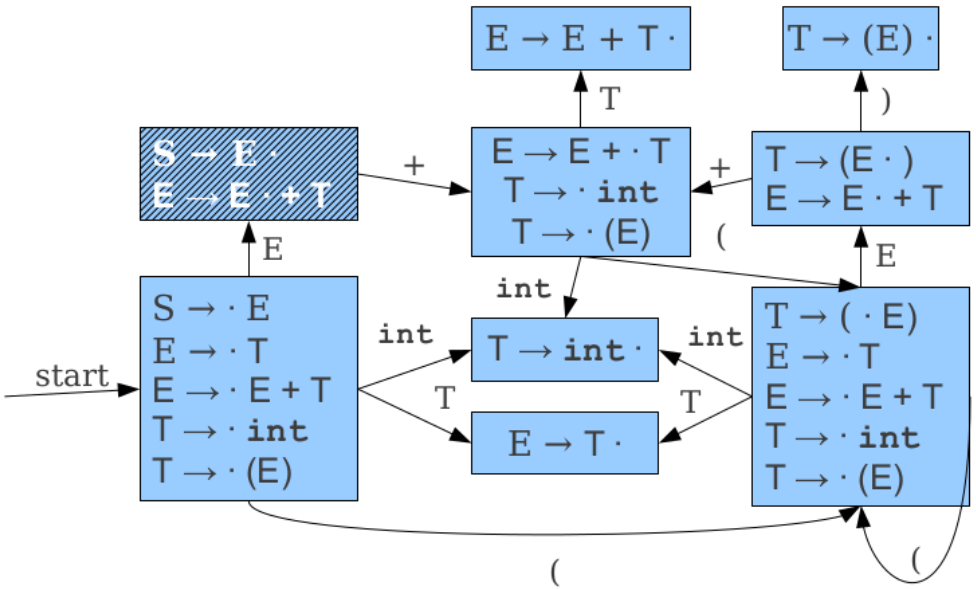
LR(1) Parsing: The Intuition

S → E
E → T
E → E + T
T → int
T → (E)



LR(1) Parsing: The Intuition

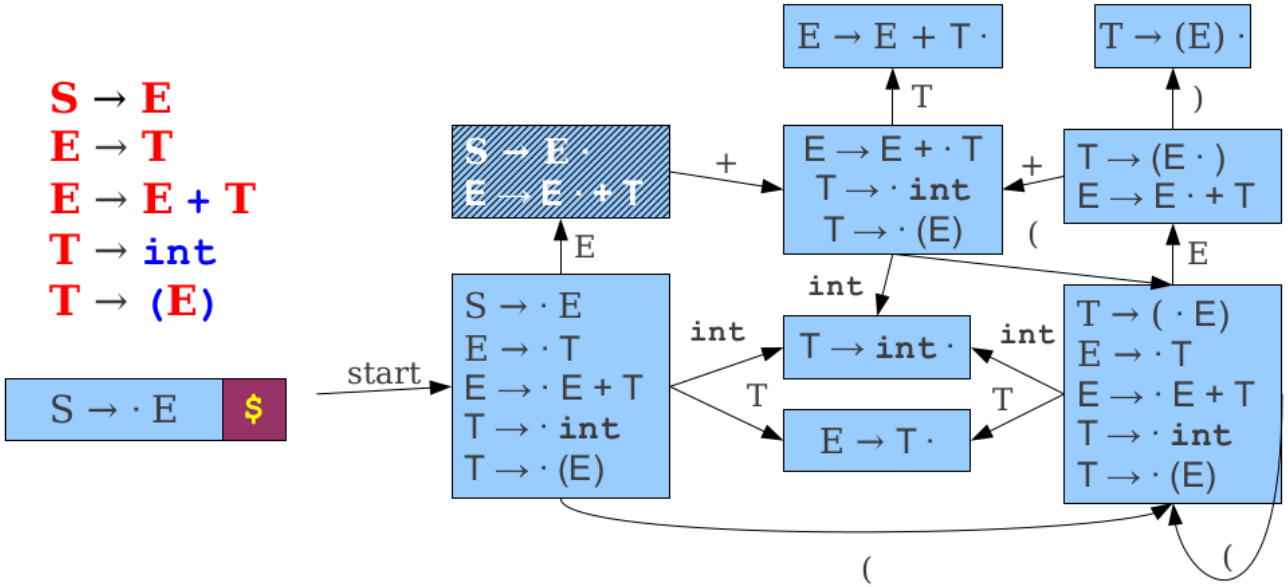
S → E
E → T
E → E + T
T → int
T → (E)



int	+	(	int	+	int	+	int	)	$

LR(1) Parsing: The Intuition

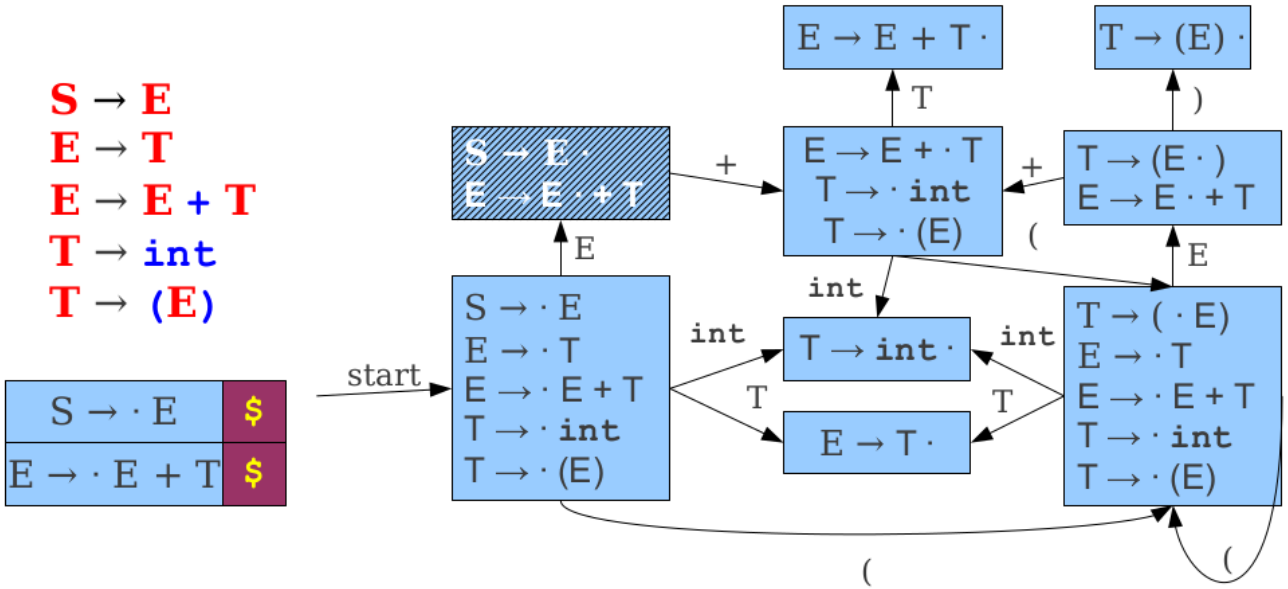
S → E
E → T
E → E + T
T → int
T → (E)



int	+	(	int	+	int	+	int	)	$

LR(1) Parsing: The Intuition

S → E
E → T
E → E + T
T → int
T → (E)

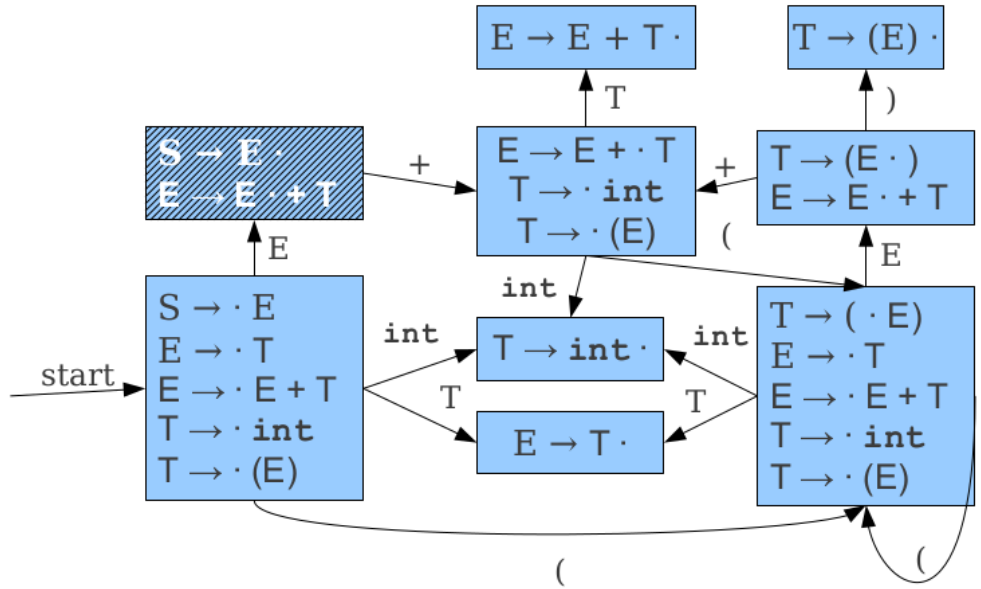


int	+	(	int	+	int	+	int	)	$

LR(1) Parsing: The Intuition

S → E
E → T
E → E + T
T → int
T → (E)

S → · E	$
E → · E + T	$
E → · T	+

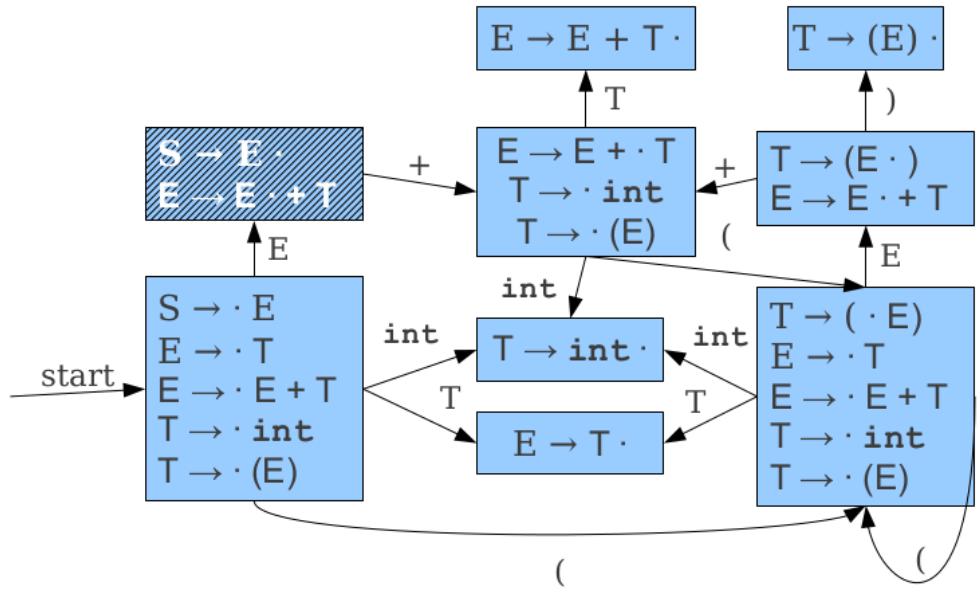


int	+	(	int	+	int	+	int	)	$

LR(1) Parsing: The Intuition

S → E
E → T
E → E + T
T → int
T → (E)

S → · E	$
E → · E + T	$
E → · T	+
T → · int	+

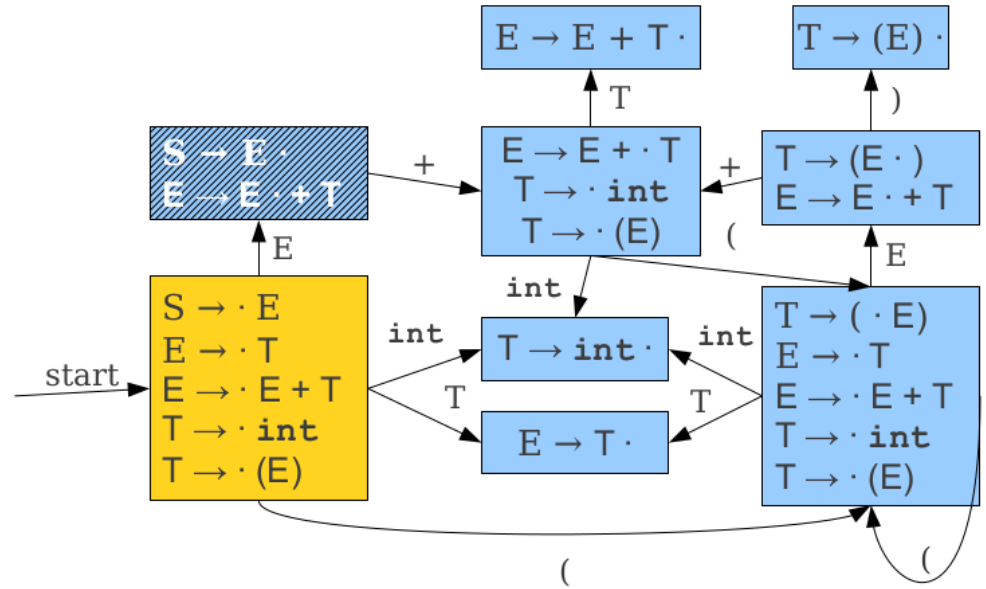


int	+	(	int	+	int	+	int	)	$

LR(1) Parsing: The Intuition

S → E
E → T
E → E + T
T → int
T → (E)

S → · E	$
E → · E + T	$
E → · T	+
T → · int	+

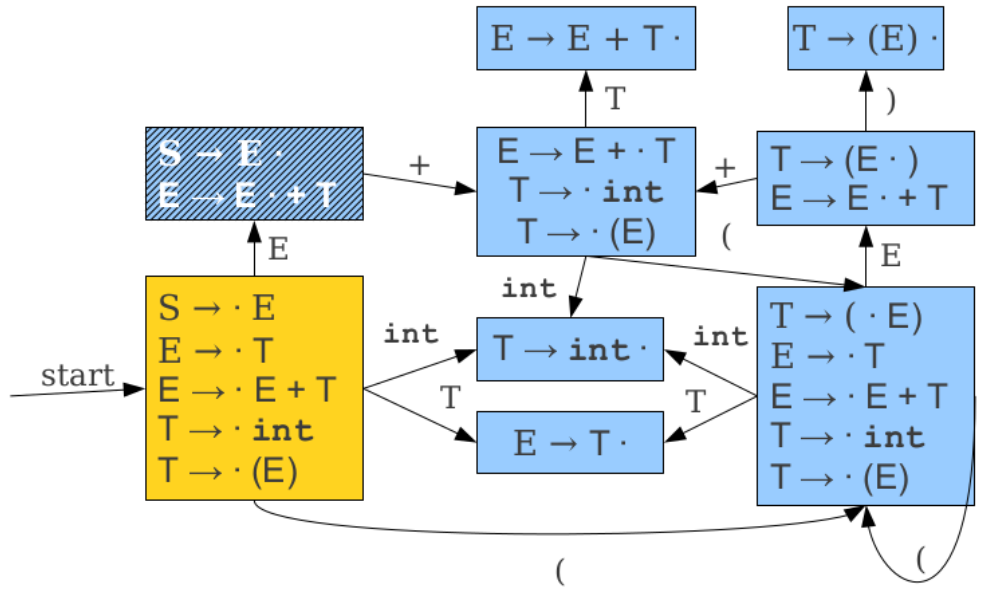


int	+	(	int	+	int	+	int	)	$

LR(1) Parsing: The Intuition

S → E
E → T
E → E + T
T → int
T → (E)

S → · E	$
E → · E + T	$
E → · T	+
T → · int	+

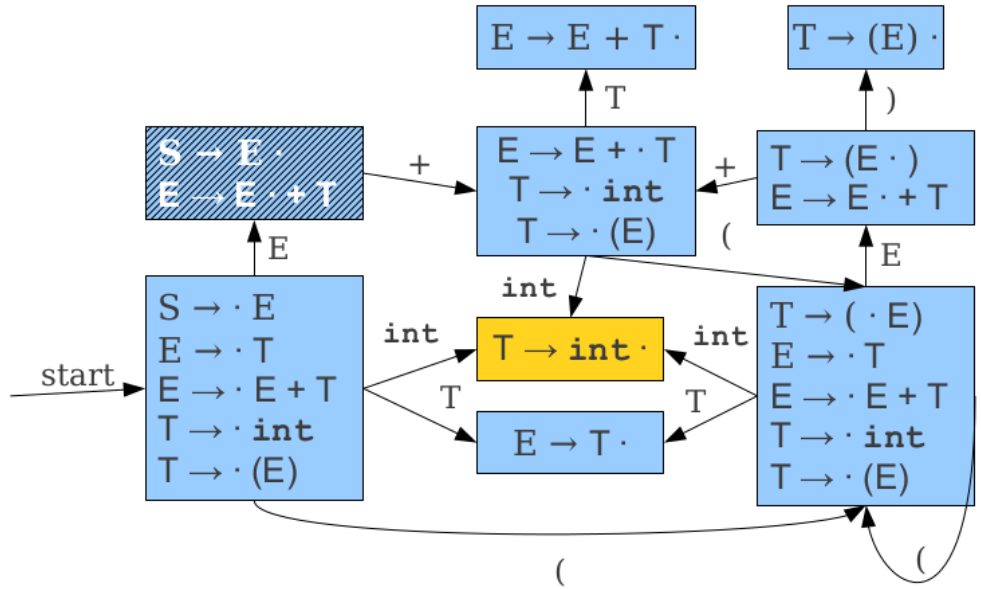


int	+	(	int	+	int	+	int	)	$

LR(1) Parsing: The Intuition

S → E
E → T
E → E + T
T → int
T → (E)

S → · E	$
E → · E + T	$
E → · T	+
T → · int	+

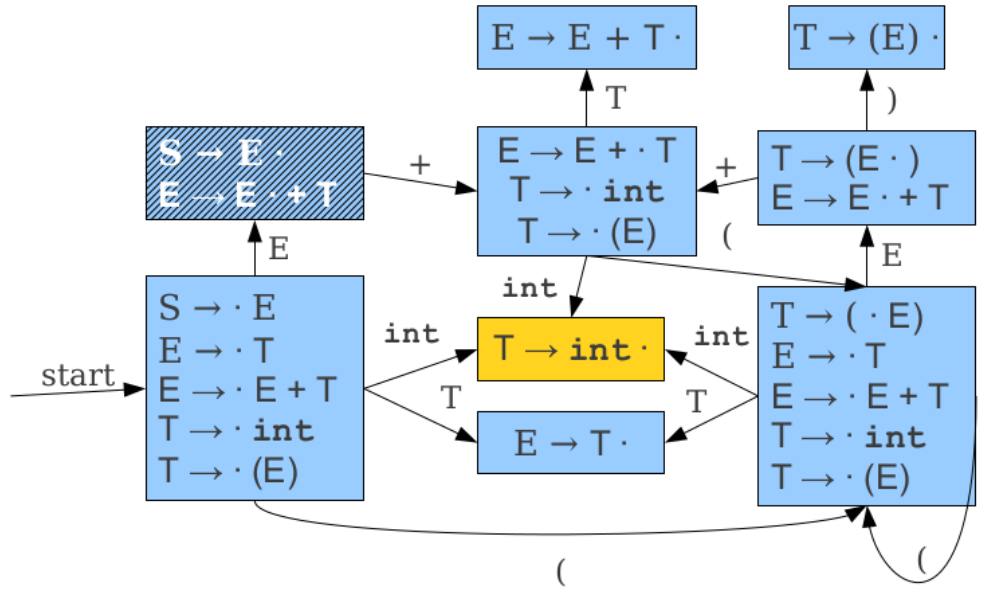


int	+	(	int	+	int	+	int	)	$

LR(1) Parsing: The Intuition

S → E
E → T
E → E + T
T → int
T → (E)

S → · E	$
E → · E + T	$
E → · T	+
T → int ·	+

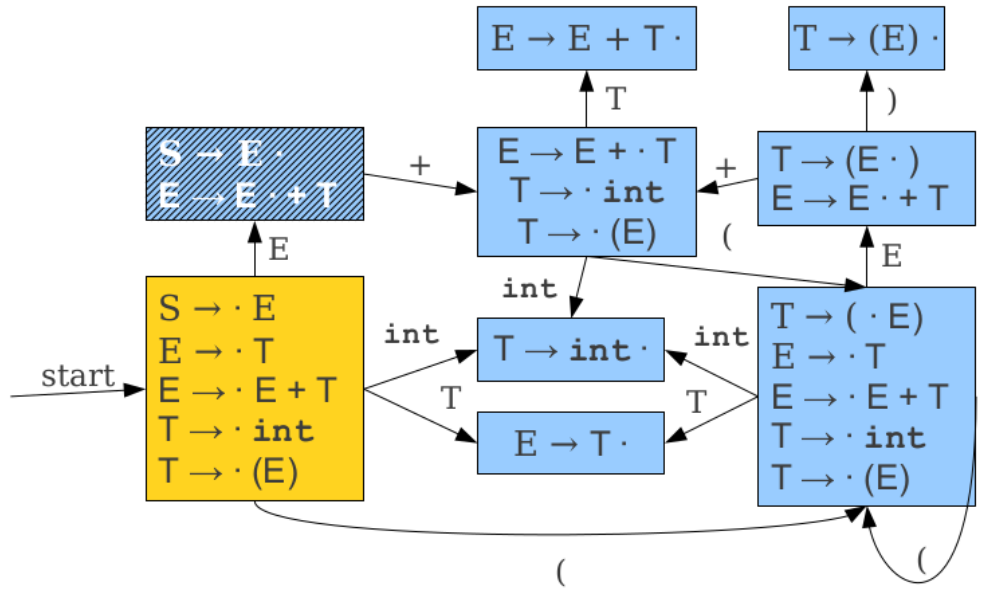


int	+	(	int	+	int	+	int	)	$

LR(1) Parsing: The Intuition

S → E
E → T
E → E + T
T → int
T → (E)

S → · E	$
E → · E + T	$
E → · T	+

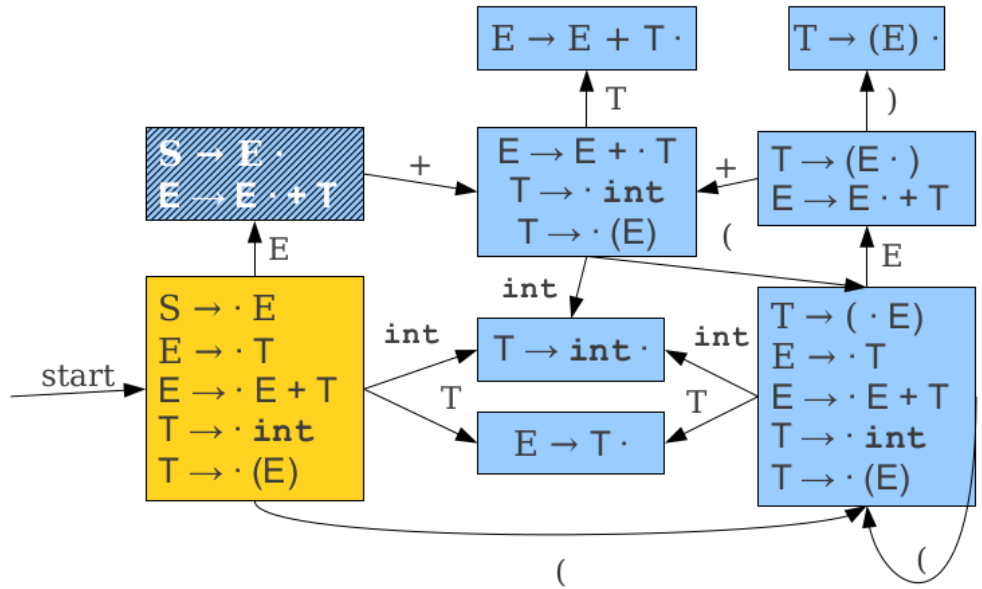


+	(	int	+	int	+	int	)	$

LR(1) Parsing: The Intuition

S → E
E → T
E → E + T
T → int
T → (E)

S → · E	$
E → · E + T	$
E → · T	+

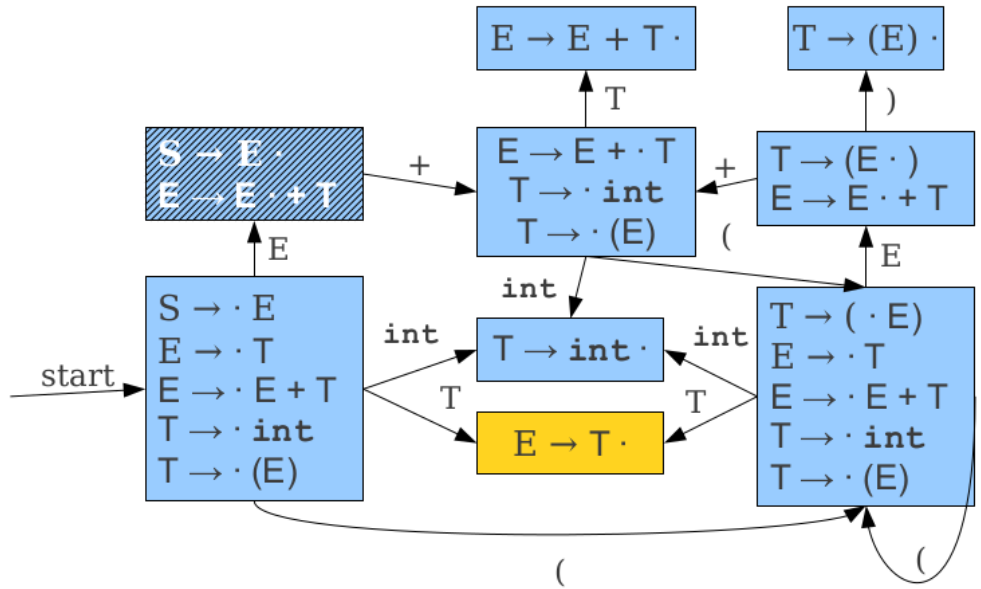


T	+	(	int	+	int	+	int	)	$

LR(1) Parsing: The Intuition

S → E
E → T
E → E + T
T → int
T → (E)

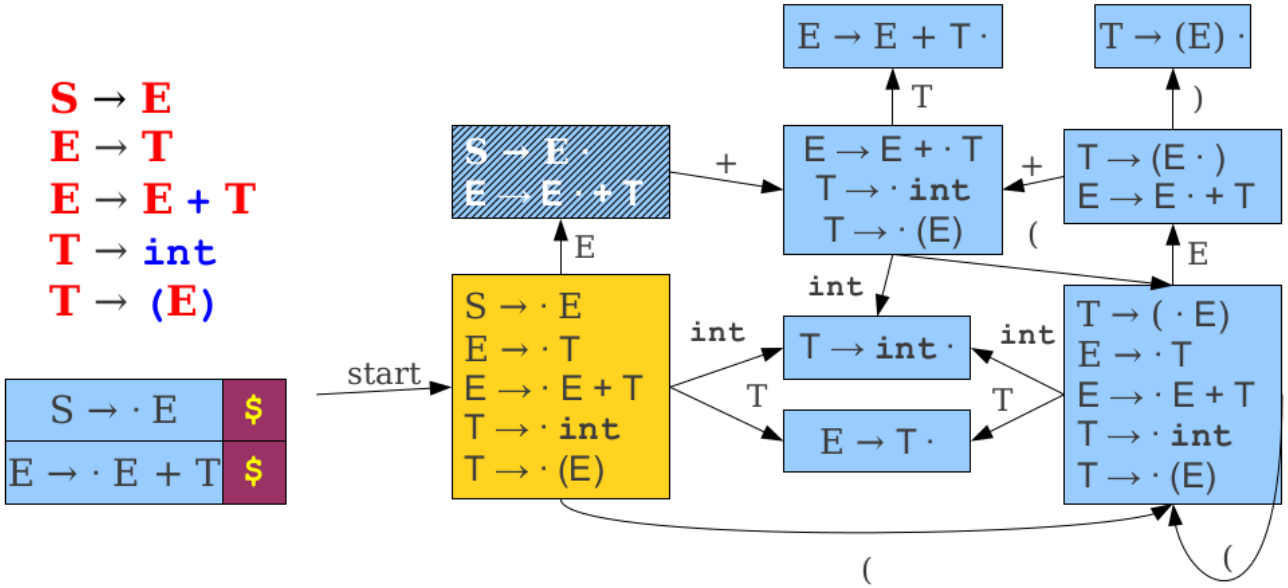
S → · E	$
E → · E + T	$
E → T ·	+



T	+	(	int	+	int	+	int	)	$

LR(1) Parsing: The Intuition

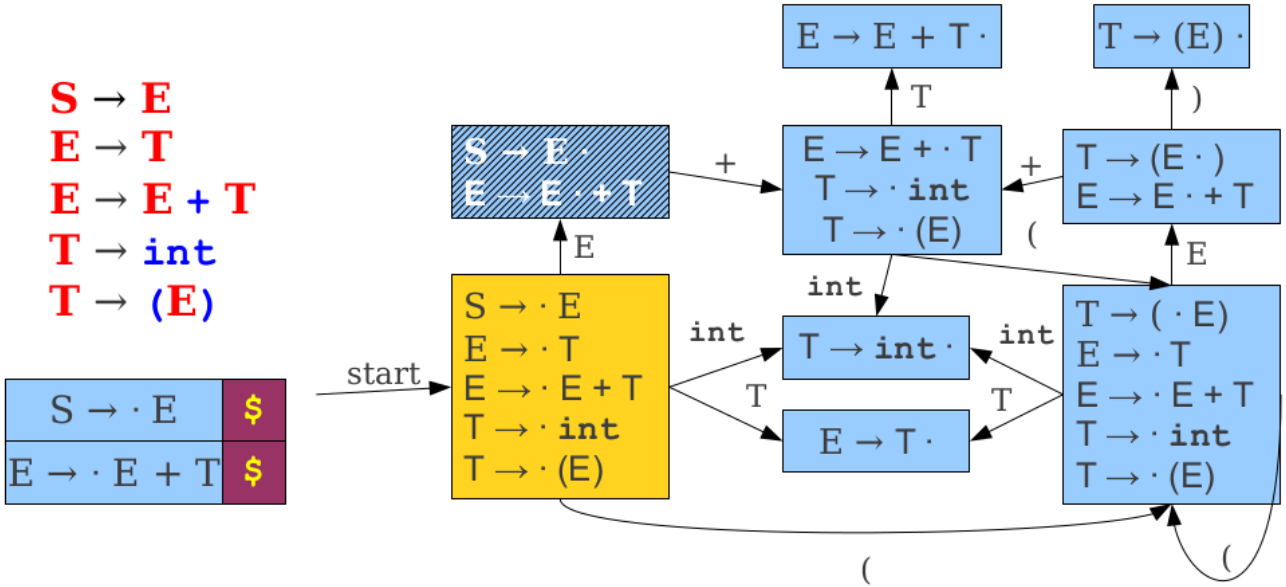
S → E
E → T
E → E + T
T → int
T → (E)



+	(	int	+	int	+	int	)	$

LR(1) Parsing: The Intuition

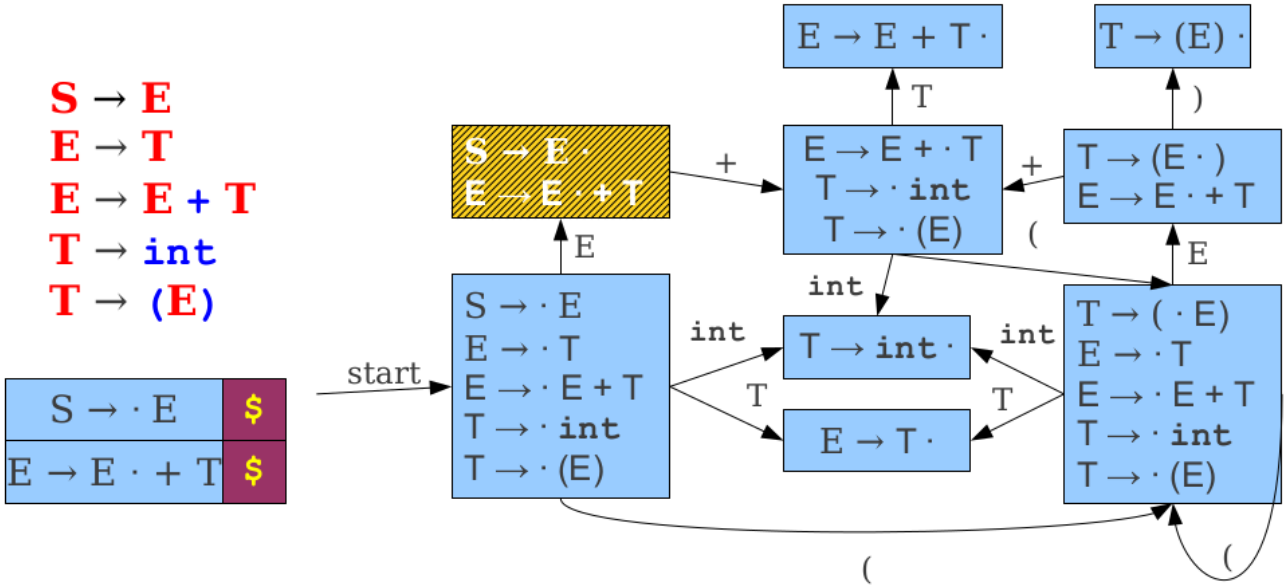
S → E
E → T
E → E + T
T → int
T → (E)



E	+	(	int	+	int	+	int	)	$

LR(1) Parsing: The Intuition

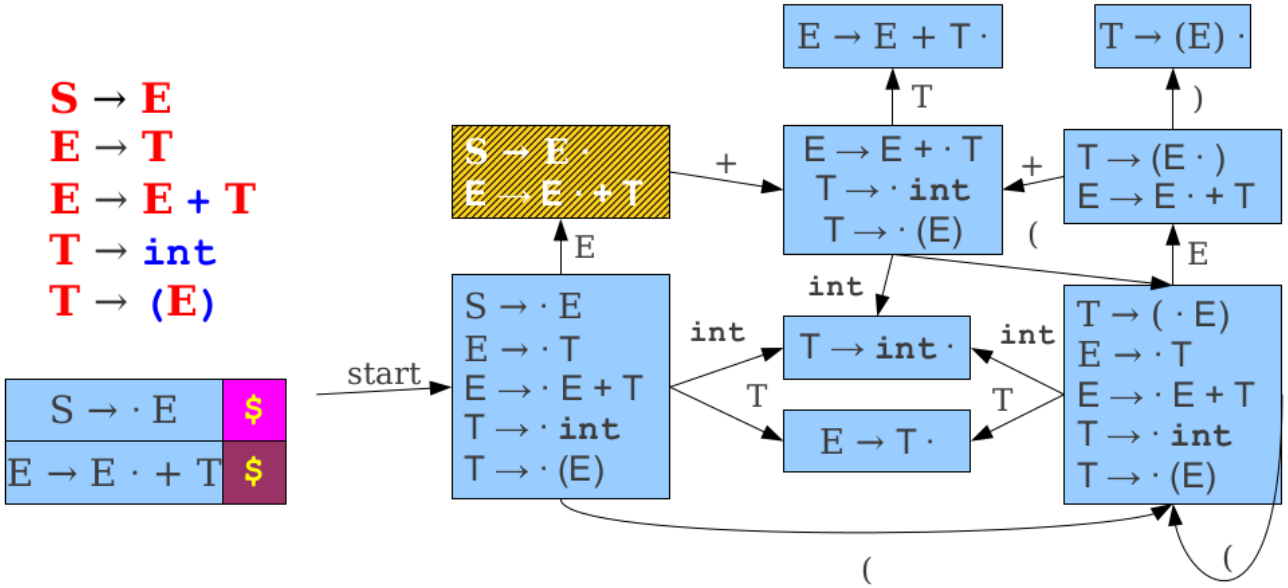
S → E
E → T
E → E + T
T → int
T → (E)

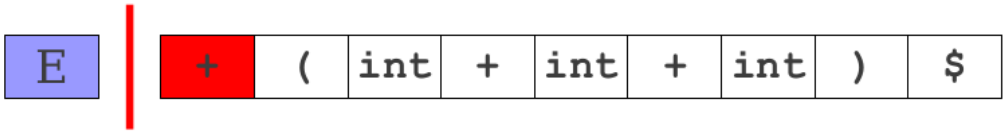


E	+	(	int	+	int	+	int	)	$

LR(1) Parsing: The Intuition

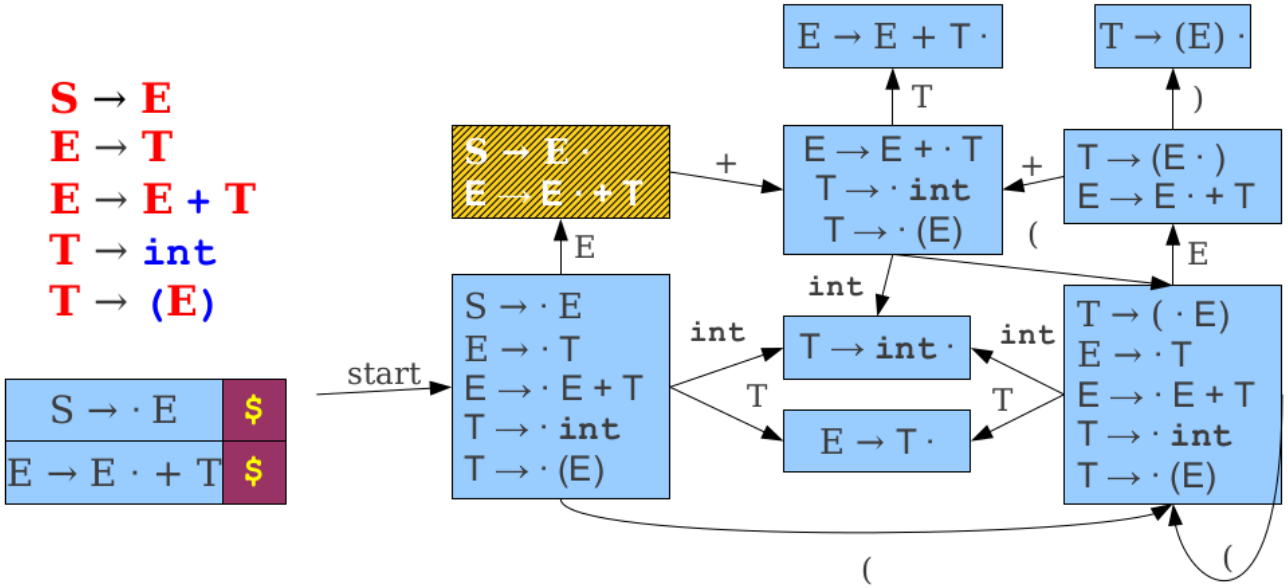
S → E
E → T
E → E + T
T → int
T → (E)





LR(1) Parsing: The Intuition

S → E
E → T
E → E + T
T → int
T → (E)

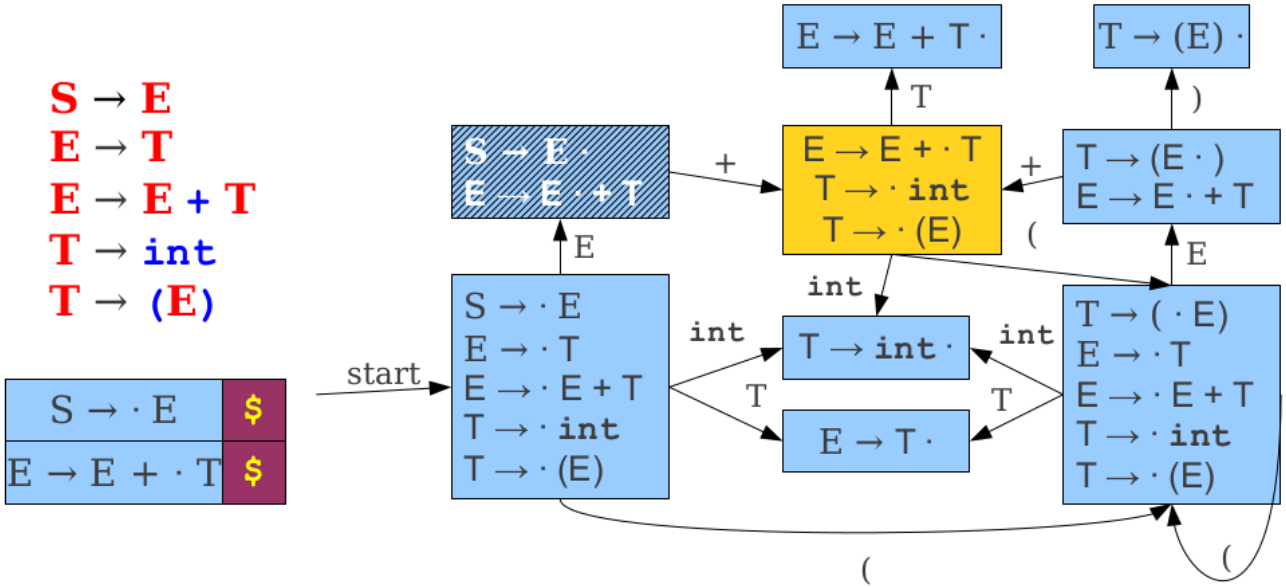


E	+

(	int	+	int	+	int	)	$

LR(1) Parsing: The Intuition

S → E
E → T
E → E + T
T → int
T → (E)

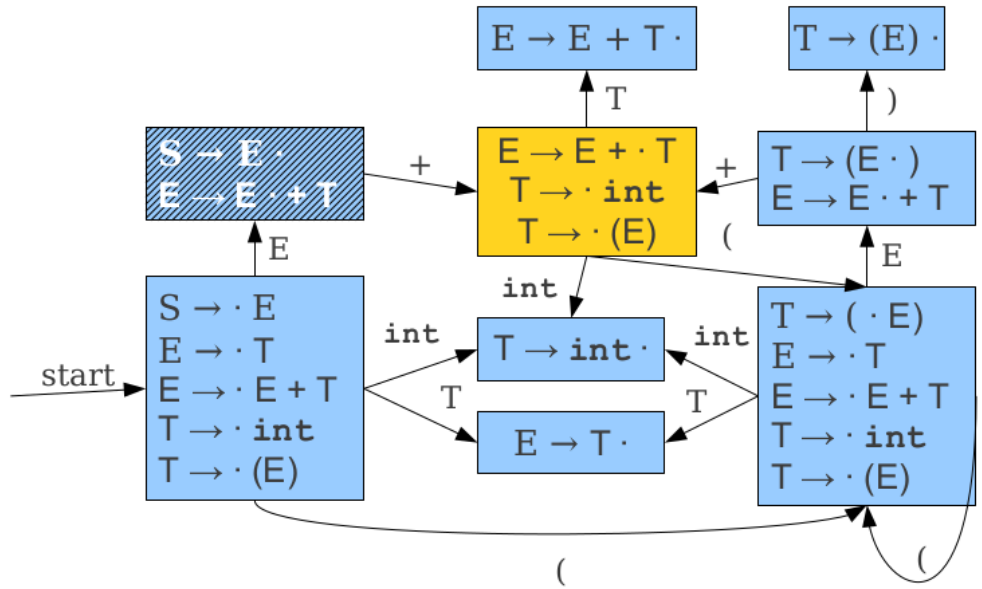


E	+	(	int	+	int	+	int	)	$

LR(1) Parsing: The Intuition

S → E
E → T
E → E + T
T → int
T → (E)

S → · E	$
E → E + · T	$
T → · (E)	$

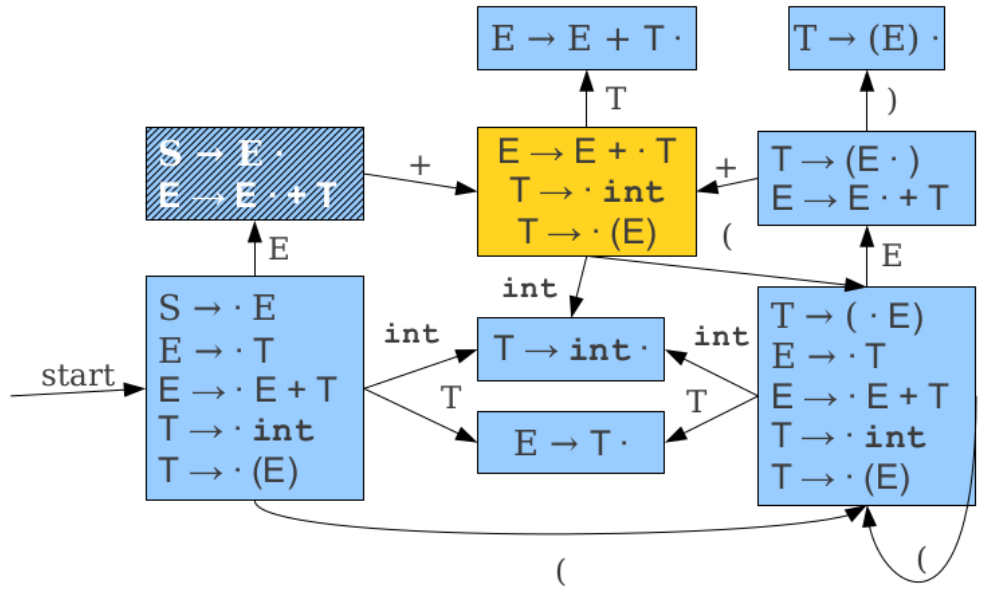


E	+	(	int	+	int	+	int	)	$

LR(1) Parsing: The Intuition

S → E
E → T
E → E + T
T → int
T → (E)

S → · E	$
E → E + · T	$
T → · (E)	$



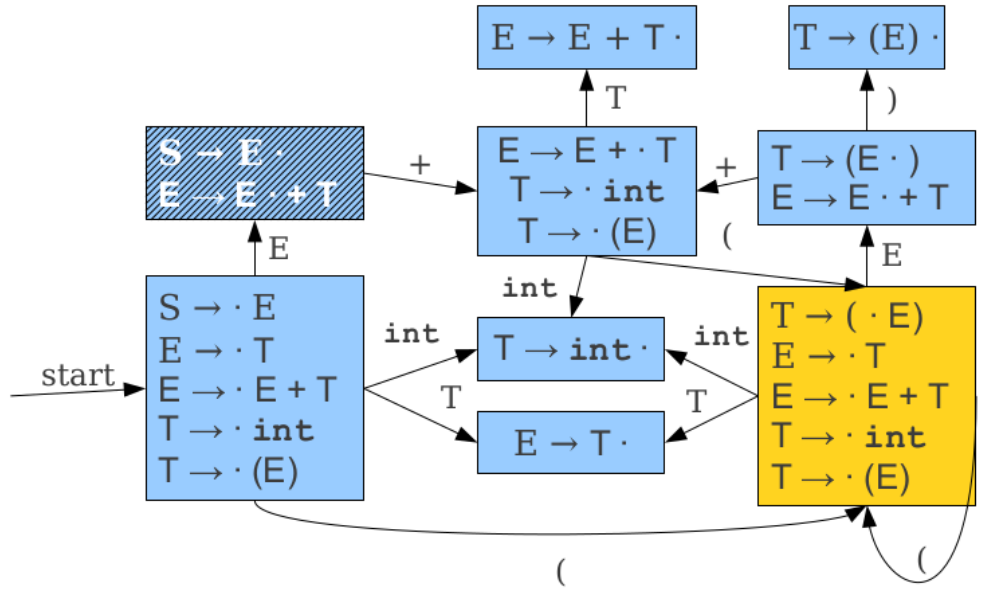
E	+	(

int	+	int	+	int	)	$

LR(1) Parsing: The Intuition

S → E
E → T
E → E + T
T → int
T → (E)

S → · E	$
E → E + · T	$
T → (· E)	$



E	+	(

int	+	int	+	int	)	$

The Intuition behind LR(1)

- Guess which series of productions we are reversing.
- Use this information to maintain information about what lookahead to expect.
- When deciding whether to shift or reduce, use lookahead to disambiguate.

Tracking Lookaheads

- How do we know what lookahead to expect at each state?
- Observation:
  - There are only finitely many productions we can be in at any point.
  - There are only finitely many positions we can be in each production.
  - There are only finitely many lookahead sets at each point.
- Construct an automaton to track lookaheads!

Constructing LR(1) Automata

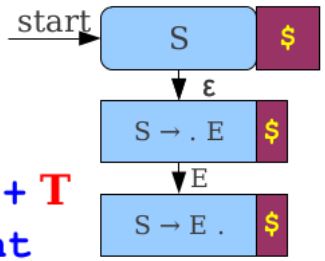
$S \rightarrow E$
$E \rightarrow T$
$E \rightarrow E + T$
$T \rightarrow \texttt{int}$
$T \rightarrow (E)$

Constructing LR(1) Automata

S → E
E → T
E → E + T
T → int
T → (E)

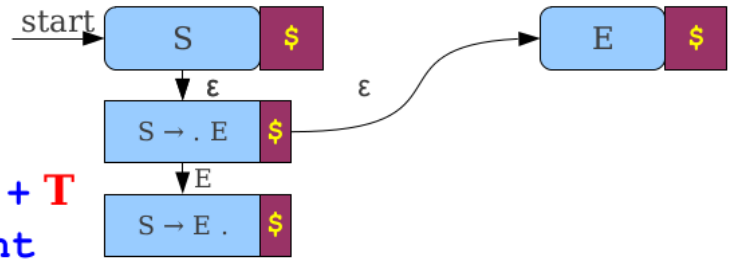
Constructing LR(1) Automata

S → E
E → T
E → E + T
T → int
T → (E)



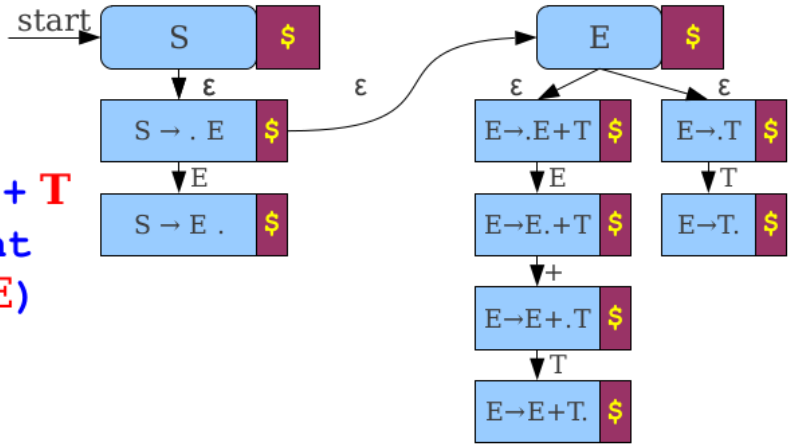
Constructing LR(1) Automata

S → E
E → T
E → E + T
T → int
T → (E)



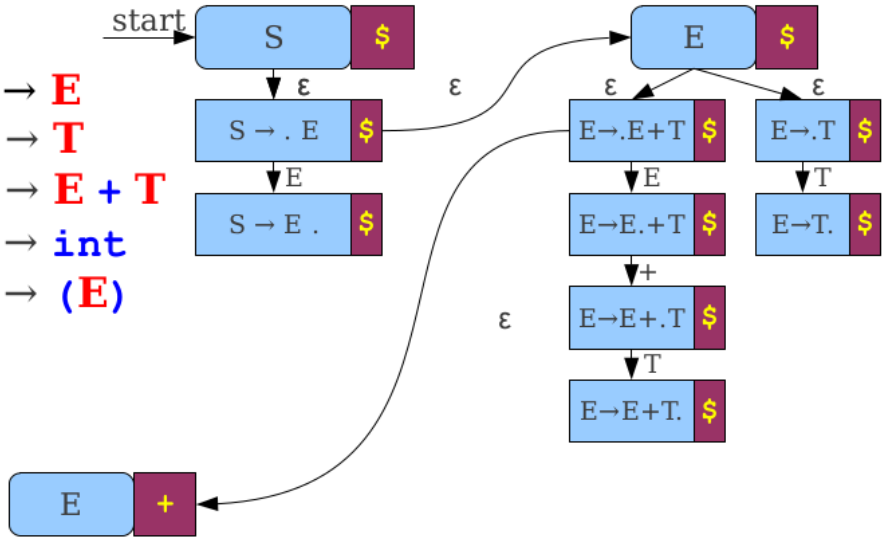
Constructing LR(1) Automata

S → E
E → T
E → E + T
T → int
T → (E)



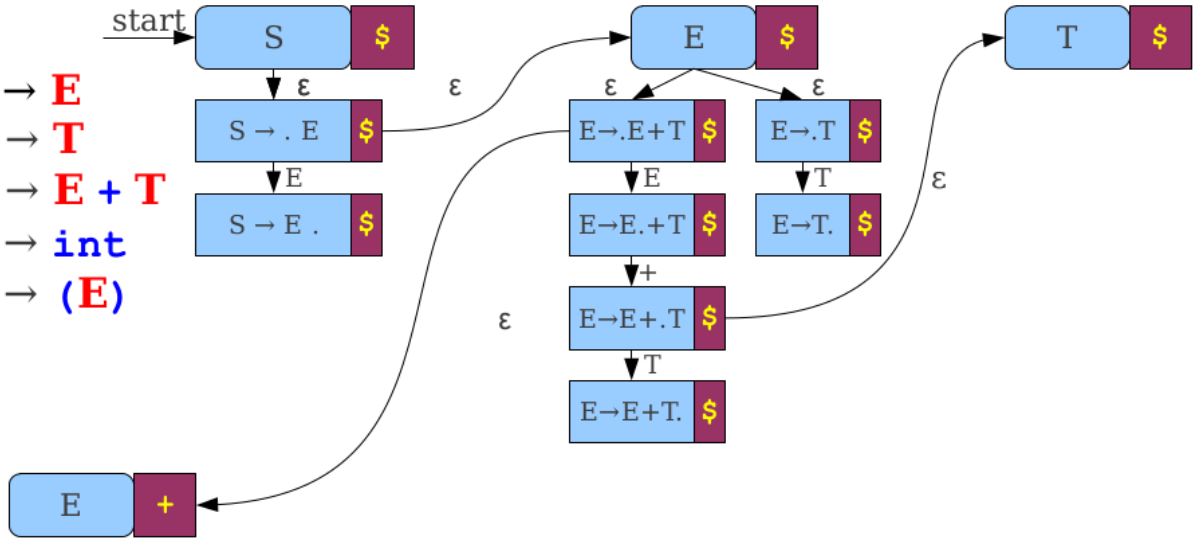
Constructing LR(1) Automata

S → E
E → T
E → E + T
T → int
T → (E)

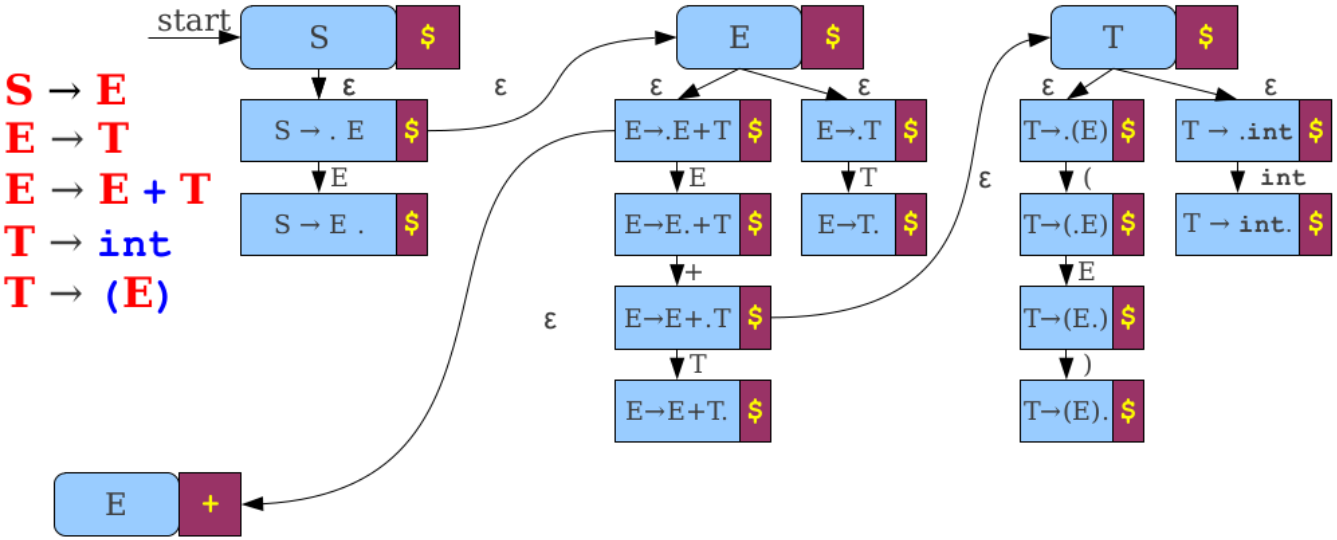


Constructing LR(1) Automata

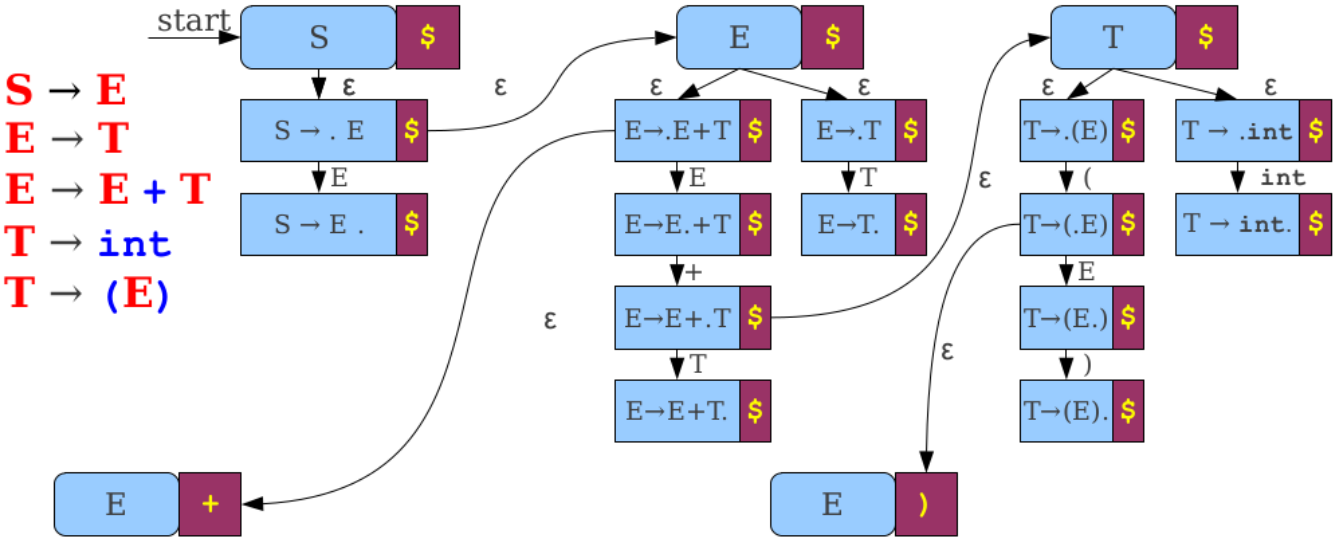
S → E
E → T
E → E + T
T → int
T → (E)



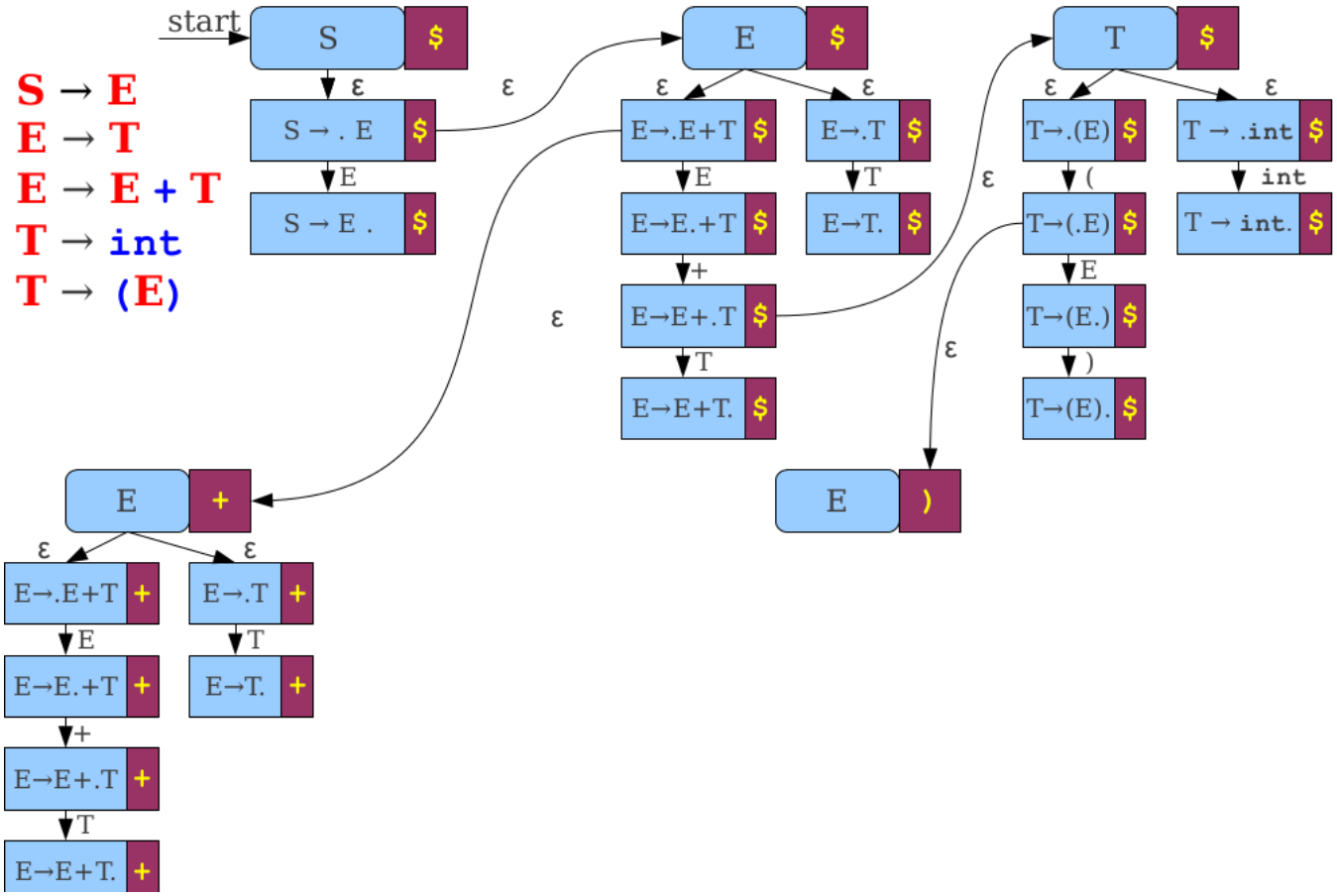
Constructing LR(1) Automata



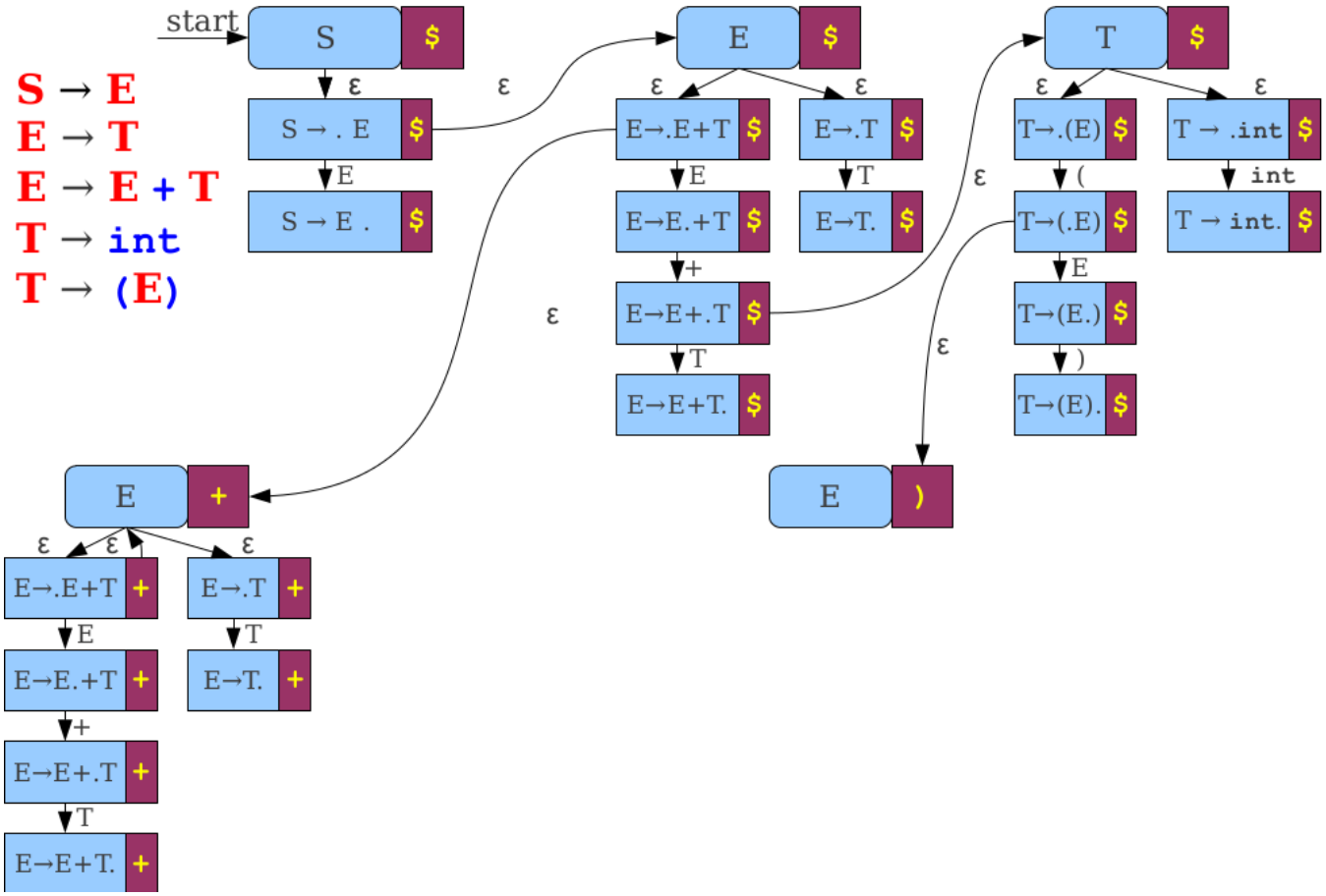
Constructing LR(1) Automata



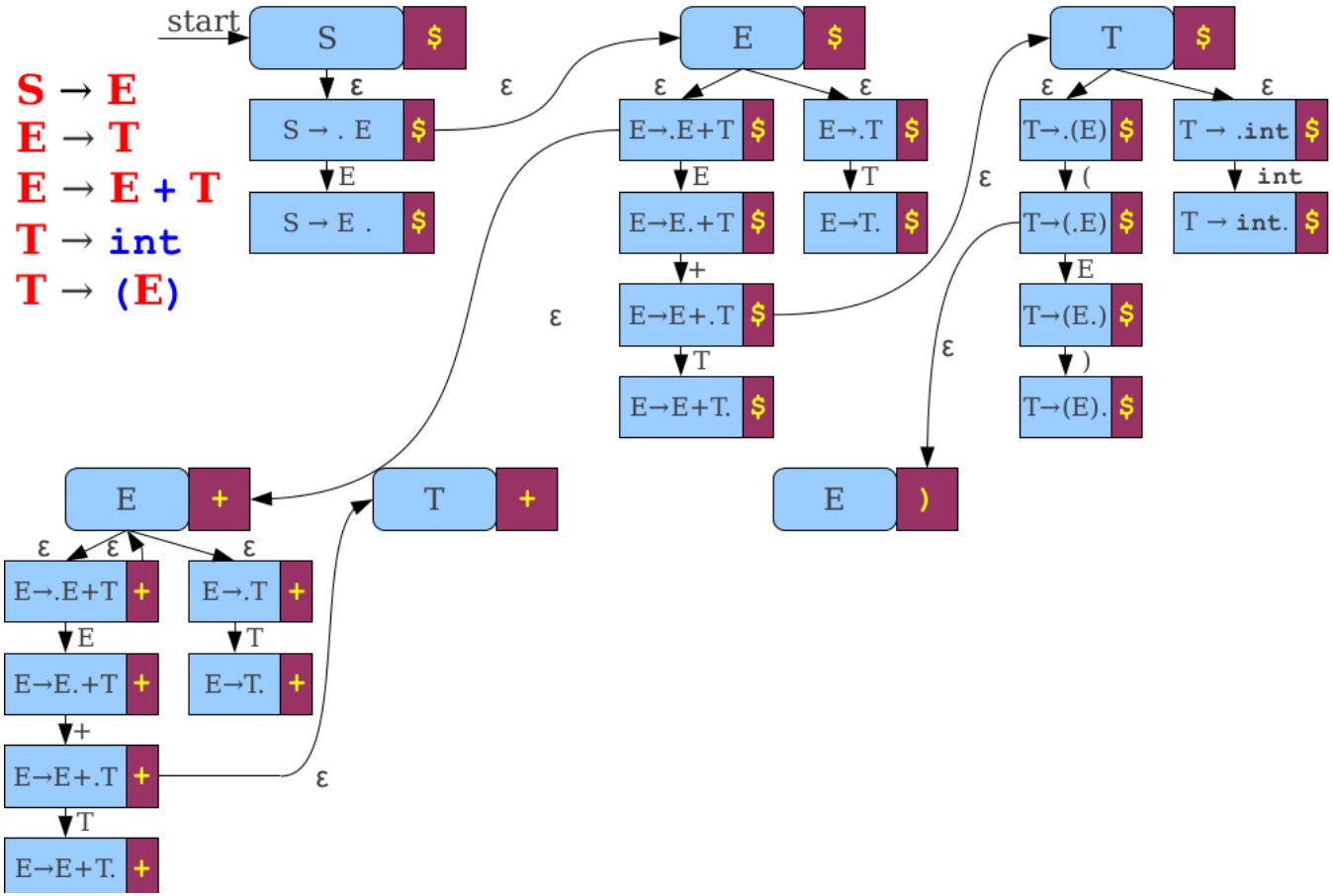
Constructing LR(1) Automata



Constructing LR(1) Automata

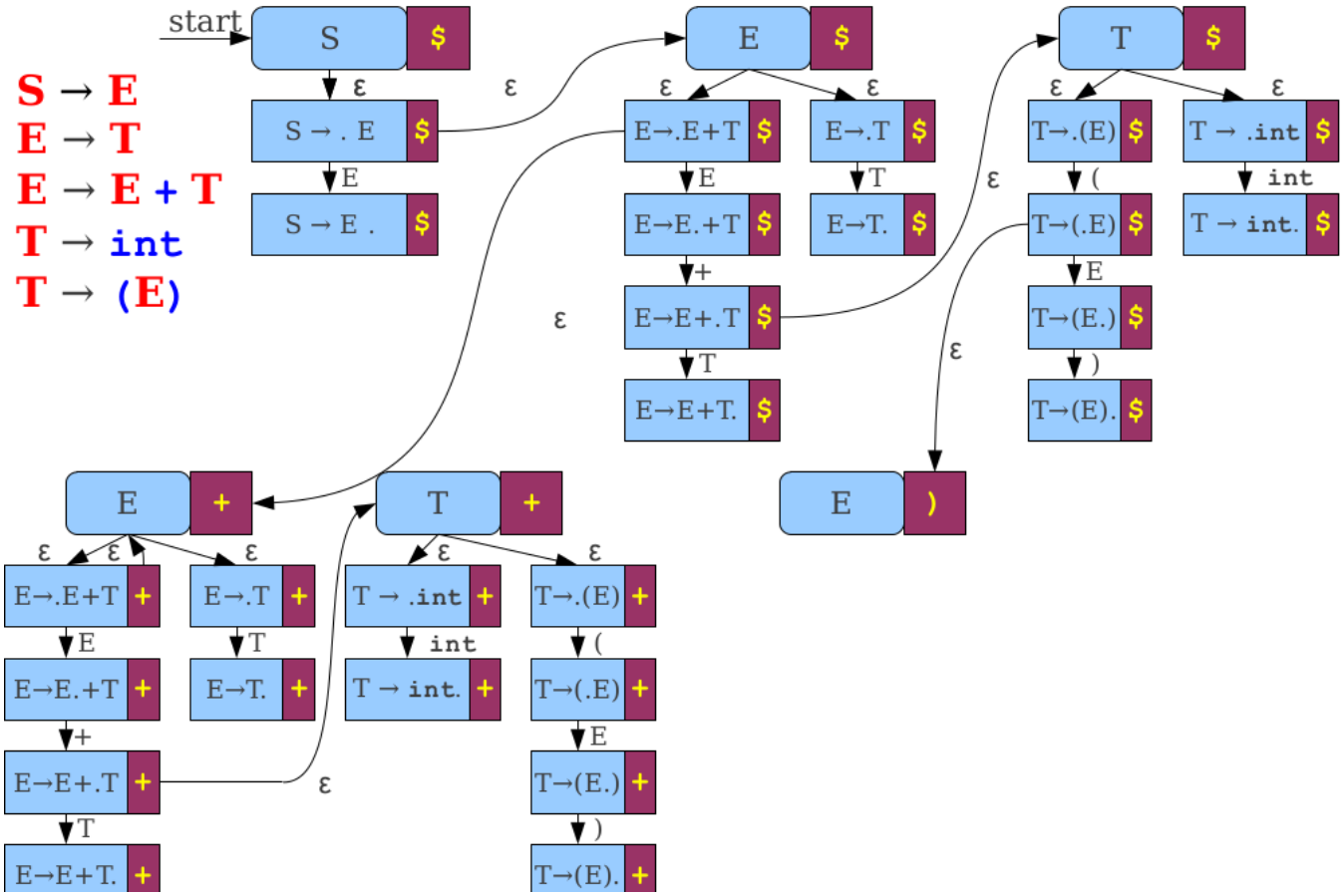


Constructing LR(1) Automata



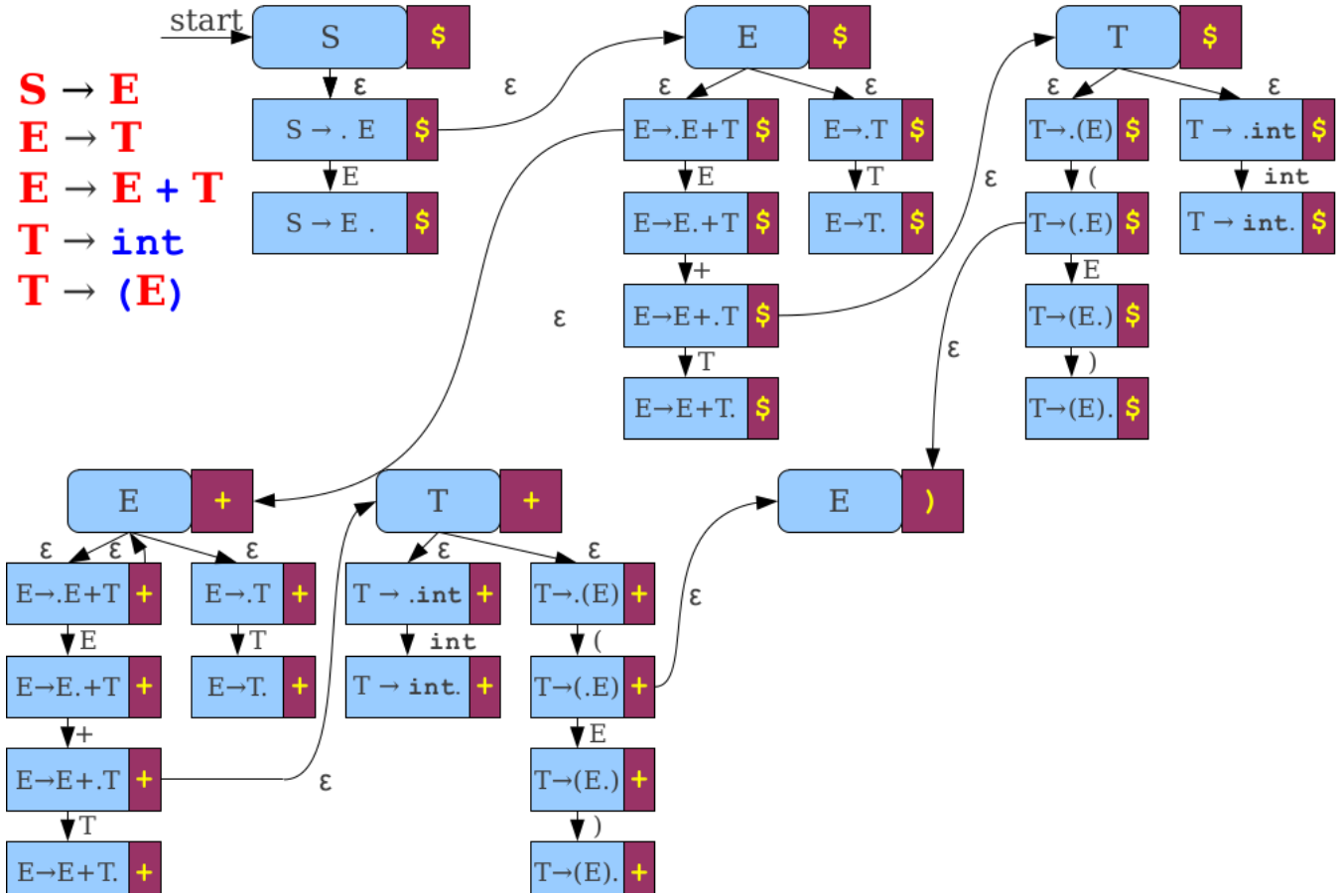
Constructing LR(1) Automata

S → E
E → T
E → E + T
T → int
T → (E)

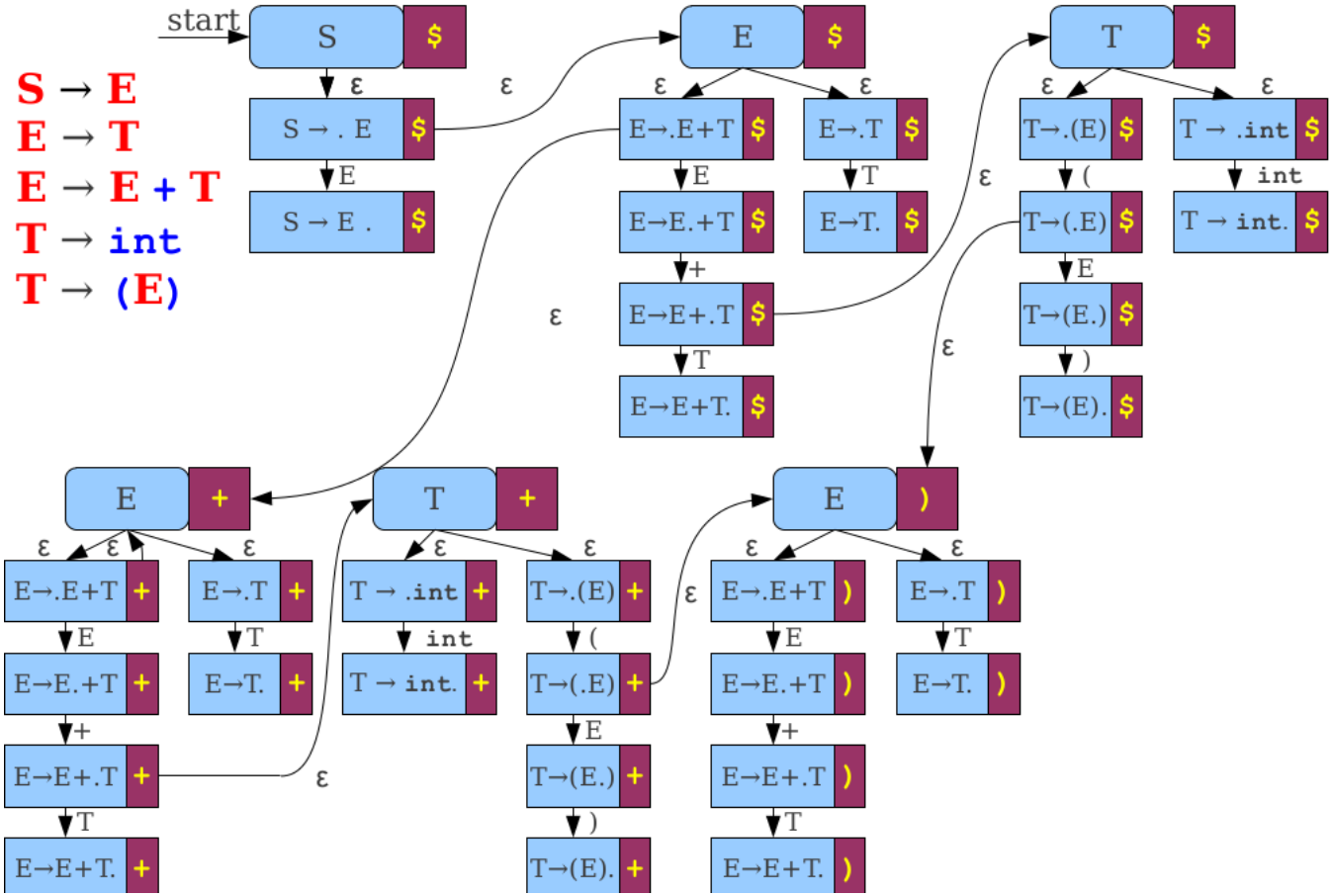


Constructing LR(1) Automata

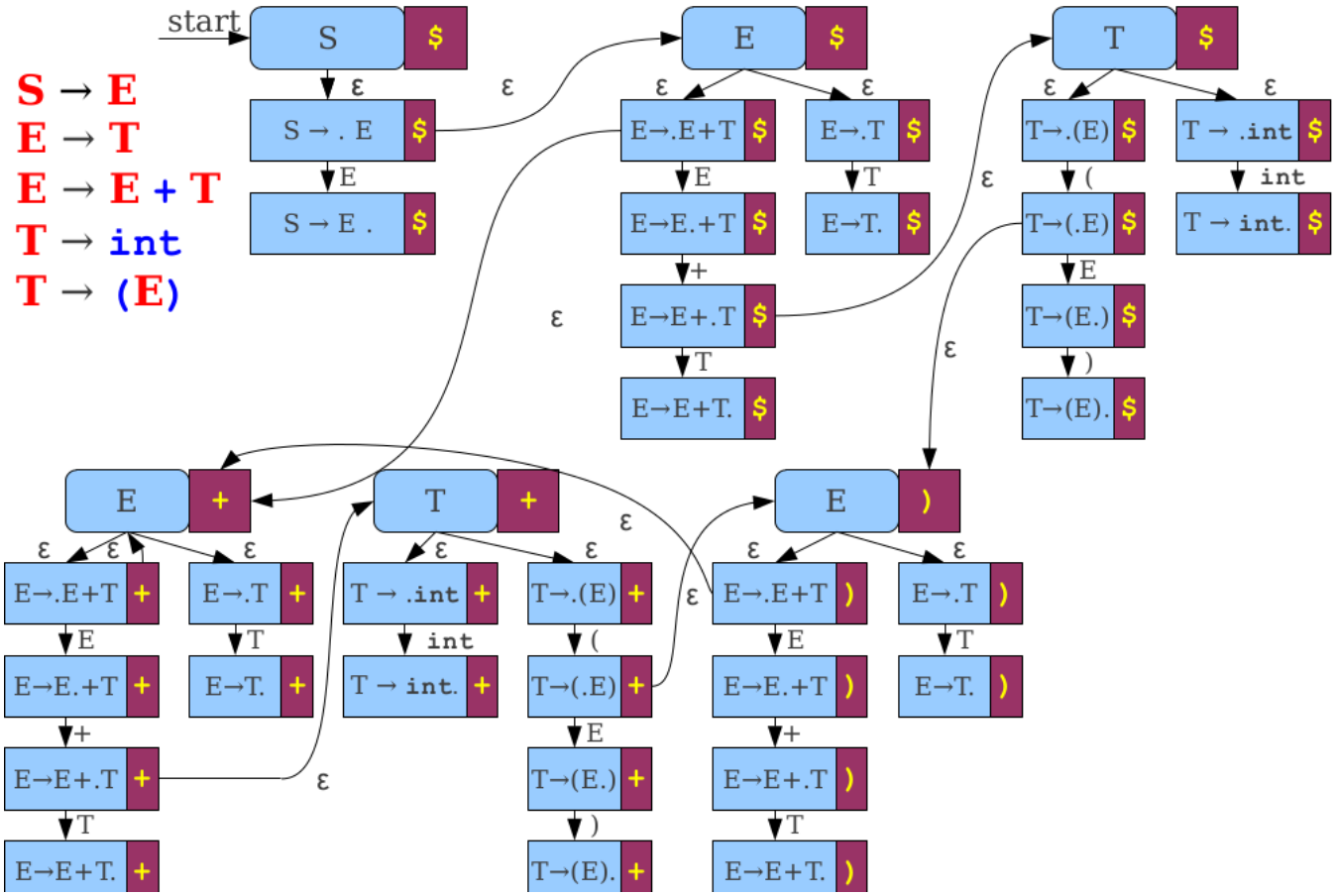
S → E
E → T
E → E + T
T → int
T → (E)



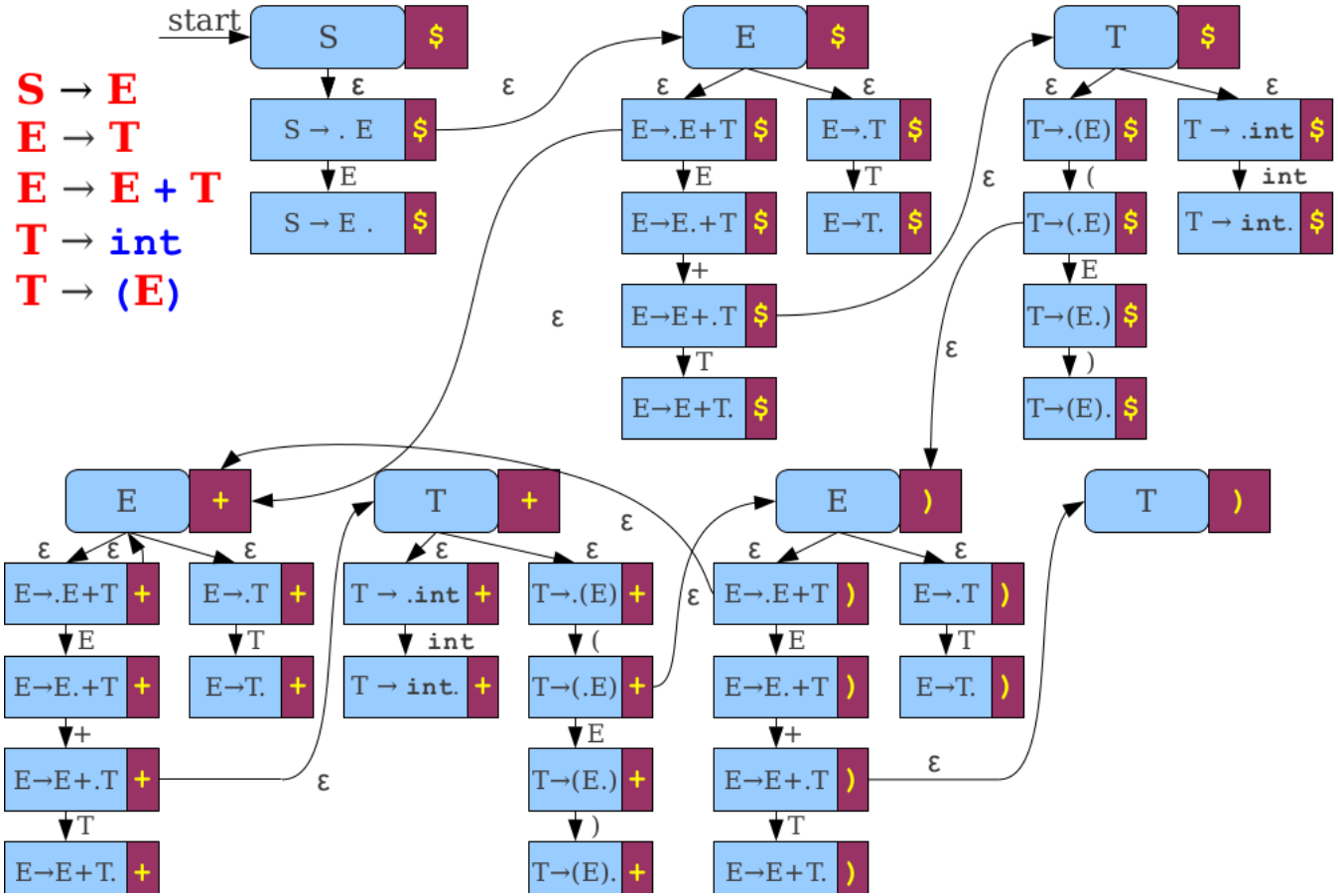
Constructing LR(1) Automata



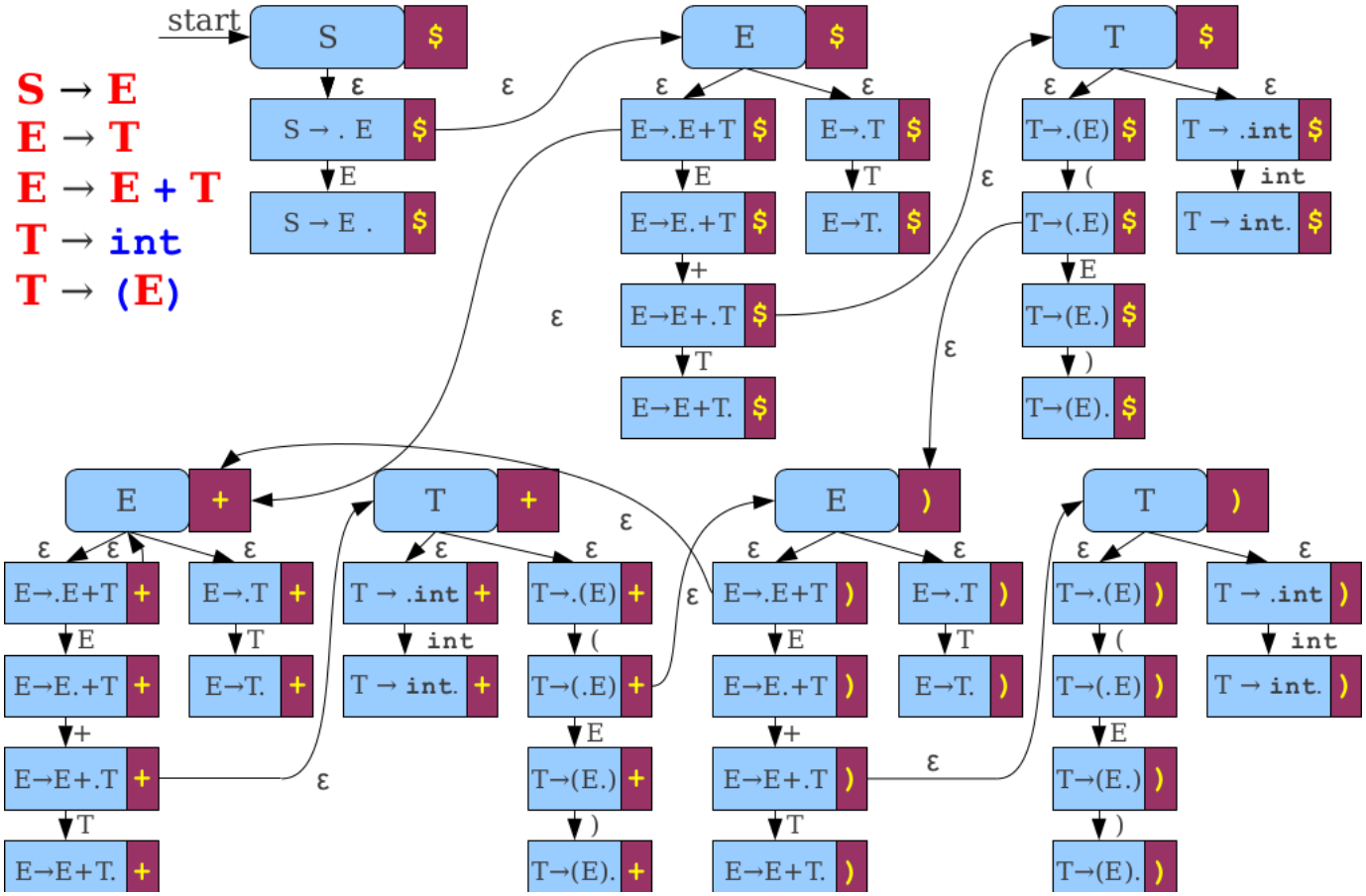
Constructing LR(1) Automata



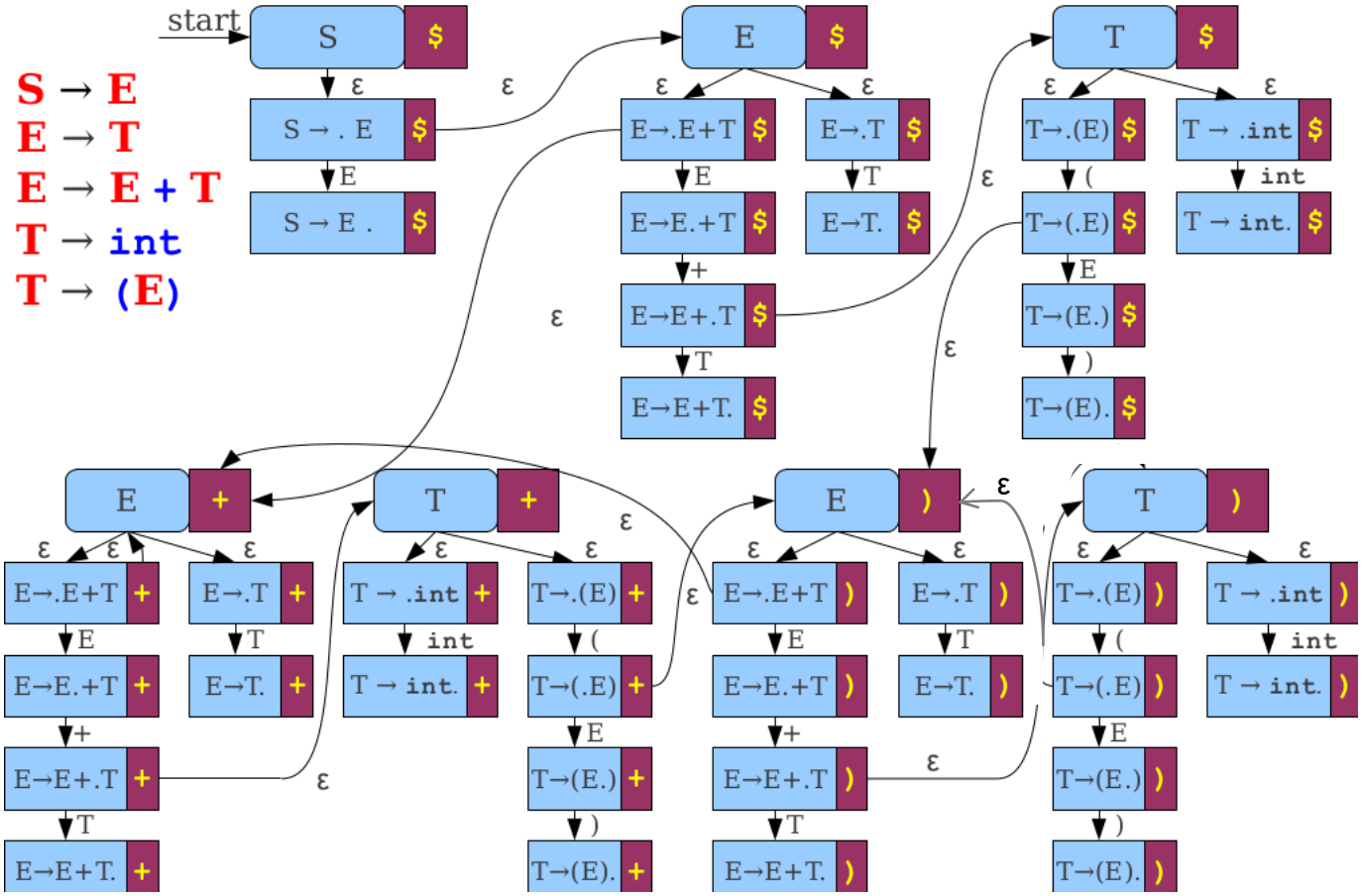
Constructing LR(1) Automata



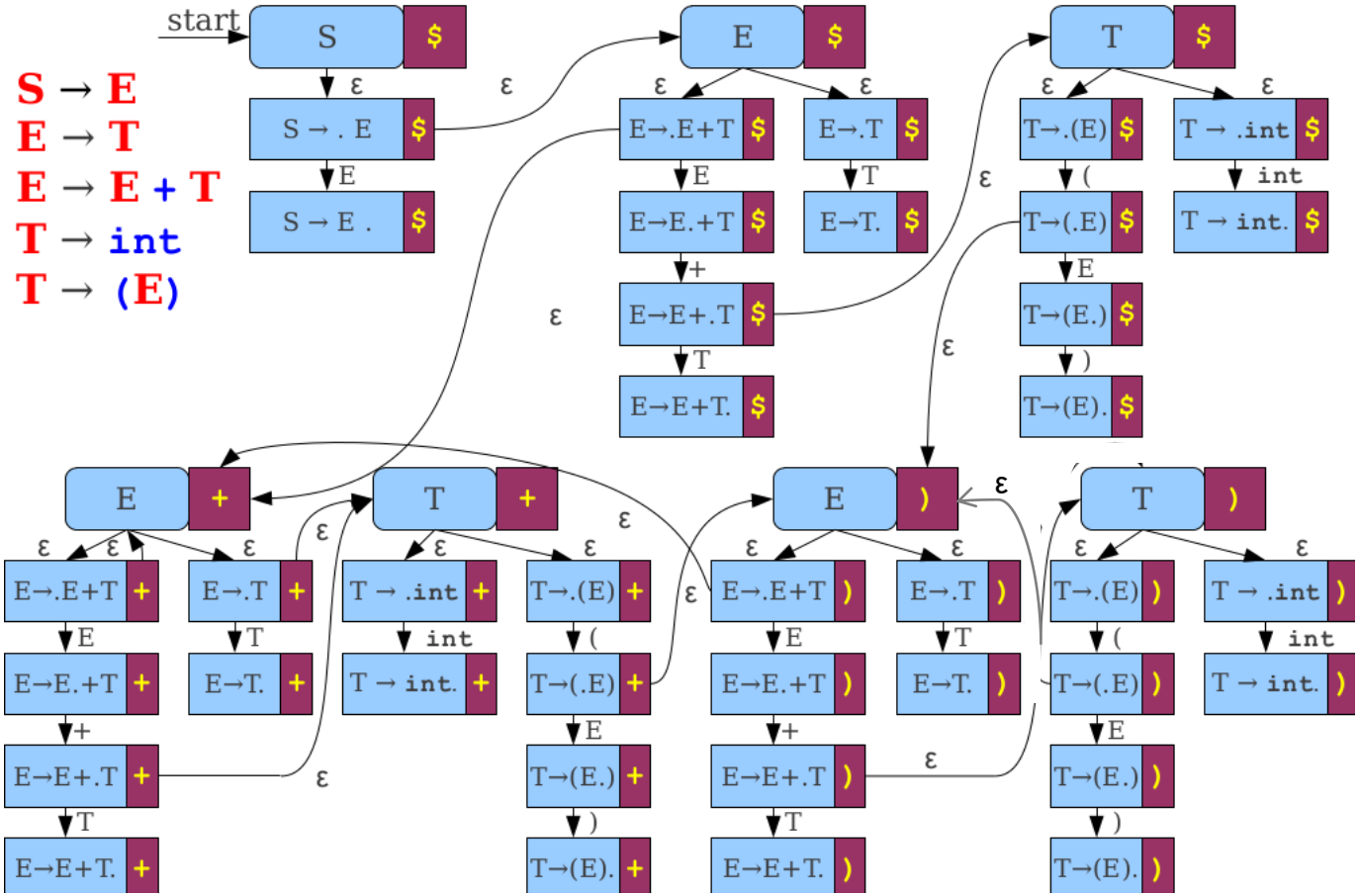
Constructing LR(1) Automata



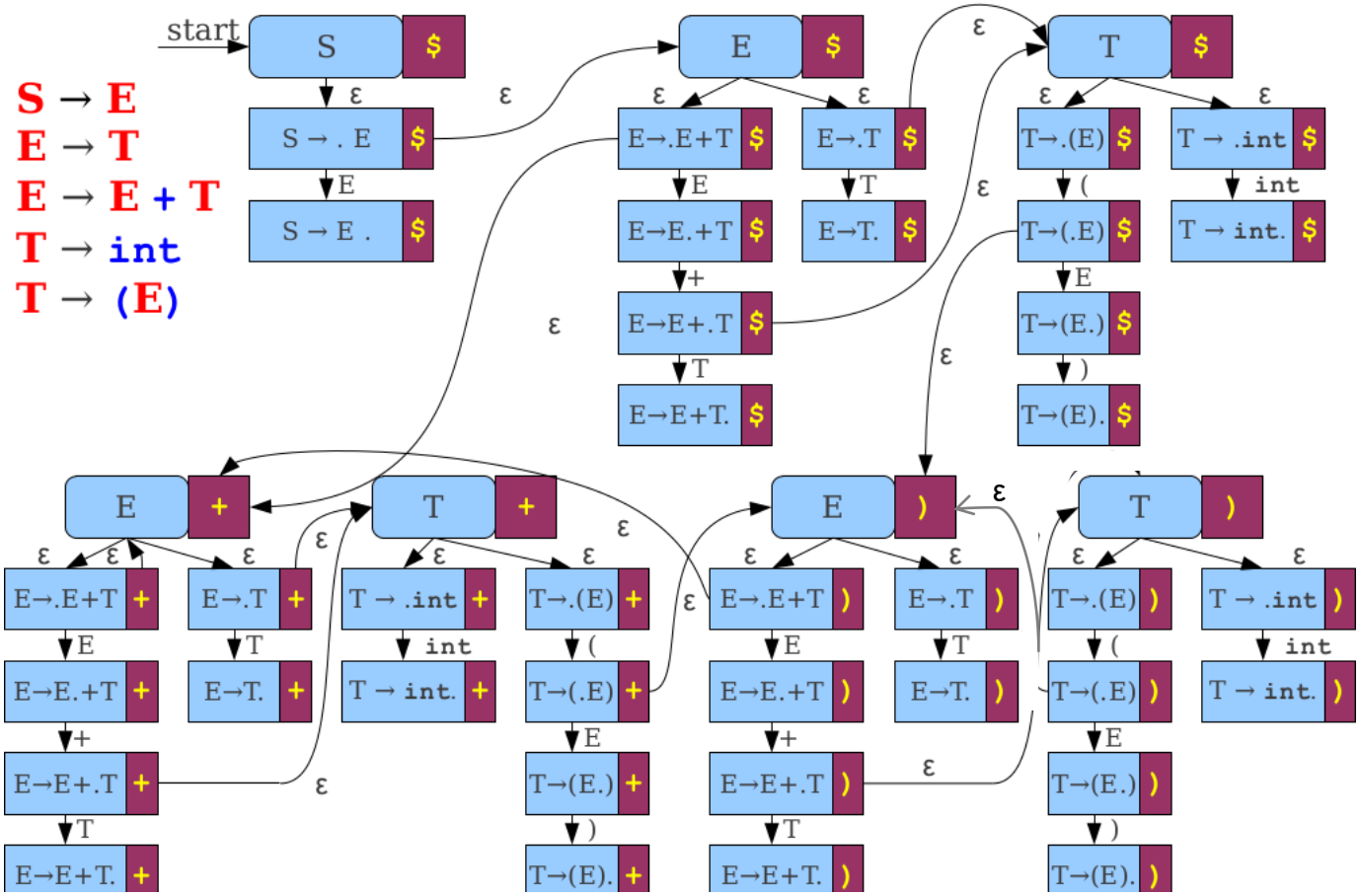
Constructing LR(1) Automata



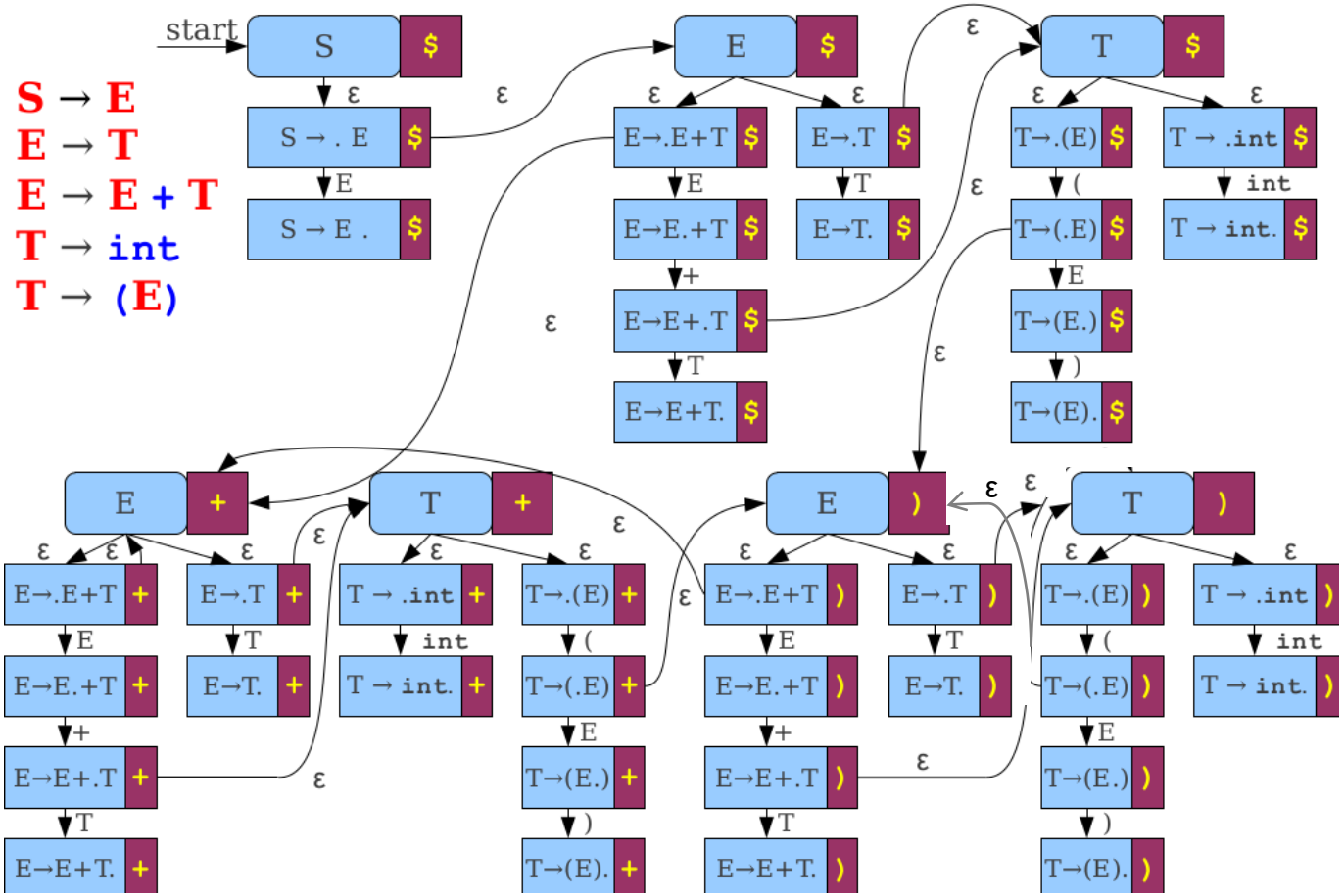
Constructing LR(1) Automata



Constructing LR(1) Automata



Constructing LR(1) Automata

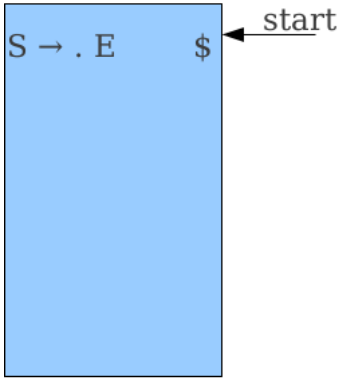


Constructing LR(1) Automata

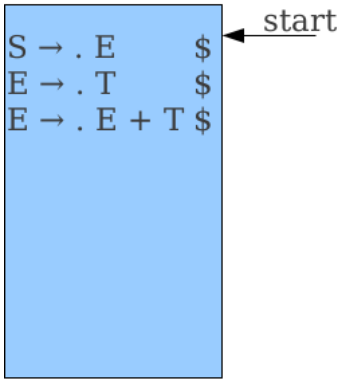
- Begin with a state S [$].
- For each state A [t], for each production $A \rightarrow \gamma$:
  - Construct states $A \rightarrow \alpha \cdot \omega$ [t] for all possible ways of splitting $\gamma = \alpha\omega$.
  - Add an ε-transition from A [t] to each of these states.
  - Add transitions on x between $A \rightarrow \alpha \cdot x\omega$ [t] and $A \rightarrow \alpha x \cdot \omega$ [t]
- For each state $A \rightarrow \alpha \cdot B\omega$ [t], add an ε-transition from $A \rightarrow \alpha \cdot B\omega$ [t] to B [r] for each terminal

  $r \in FIRST^*(\omega t)$.

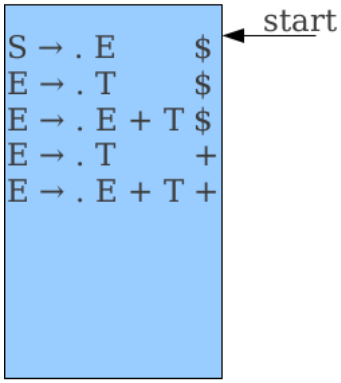
Deterministic LR(1) Automata



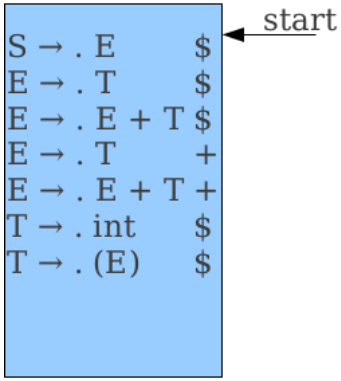
Deterministic LR(1) Automata



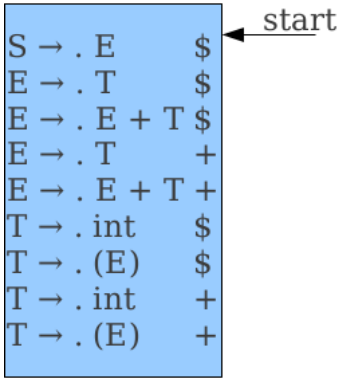
Deterministic LR(1) Automata



Deterministic LR(1) Automata



Deterministic LR(1) Automata

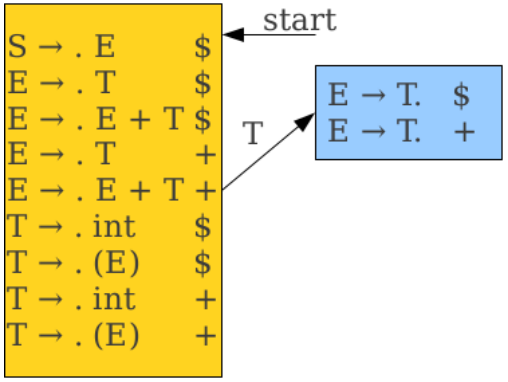


Deterministic LR(1) Automata

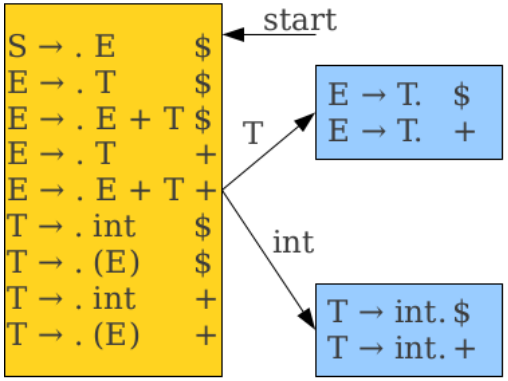
start

S → . E	$
E → . T	$
E → . E + T	$
E → . T	+
E → . E + T	+
T → . int	$
T → . (E)	$
T → . int	+
T → . (E)	+

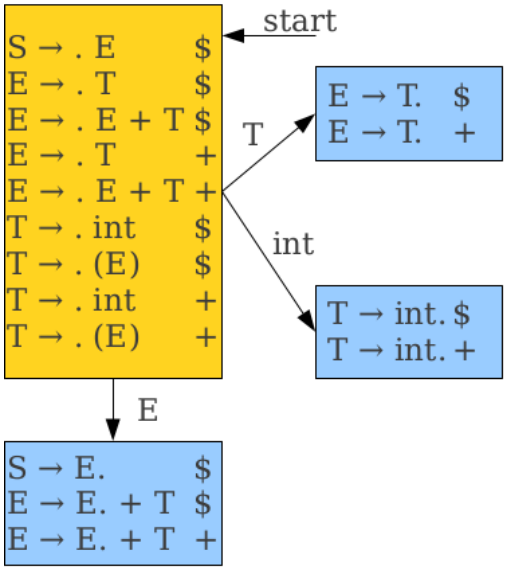
Deterministic LR(1) Automata



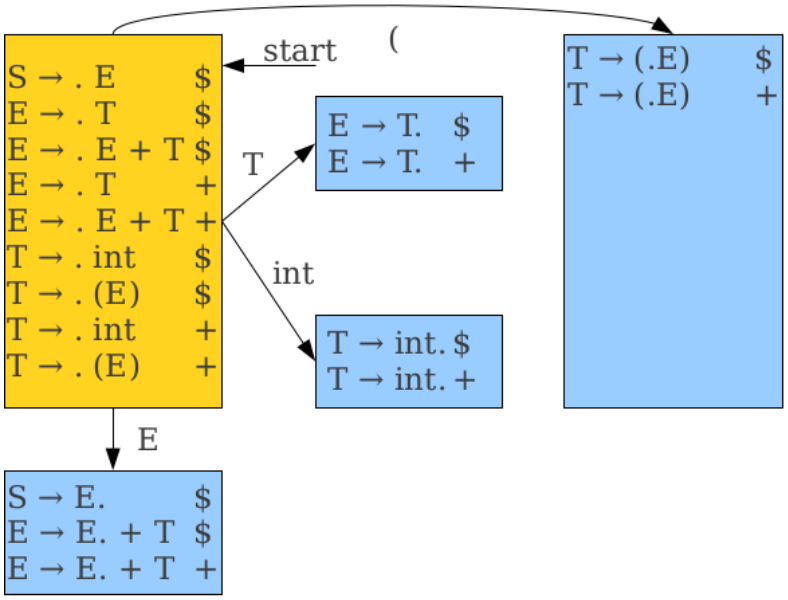
Deterministic LR(1) Automata



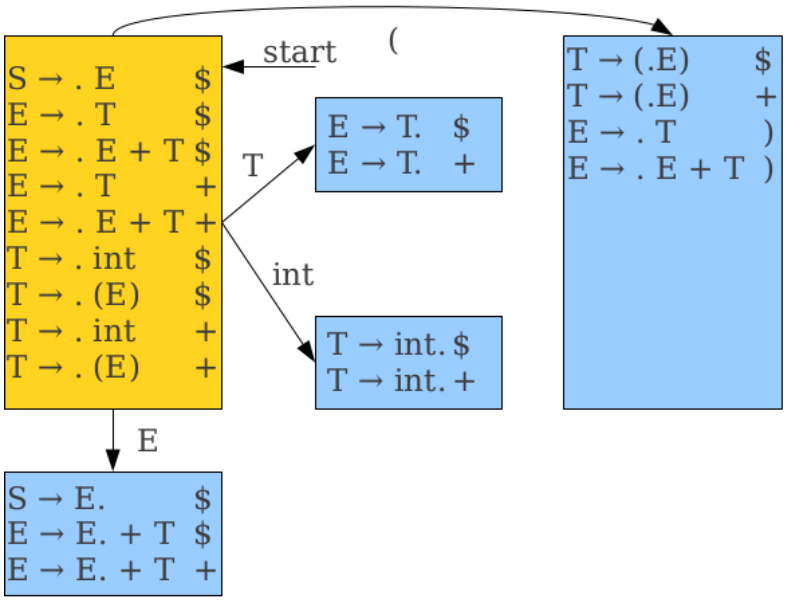
Deterministic LR(1) Automata



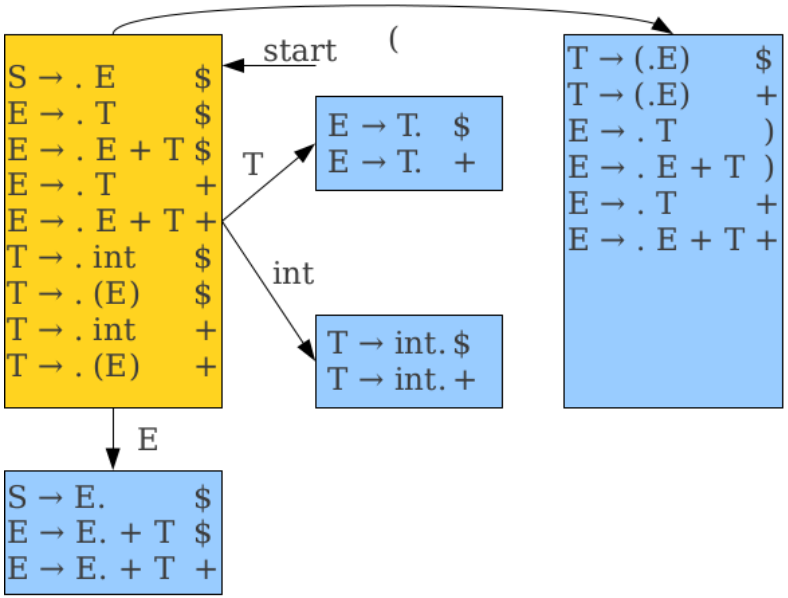
Deterministic LR(1) Automata



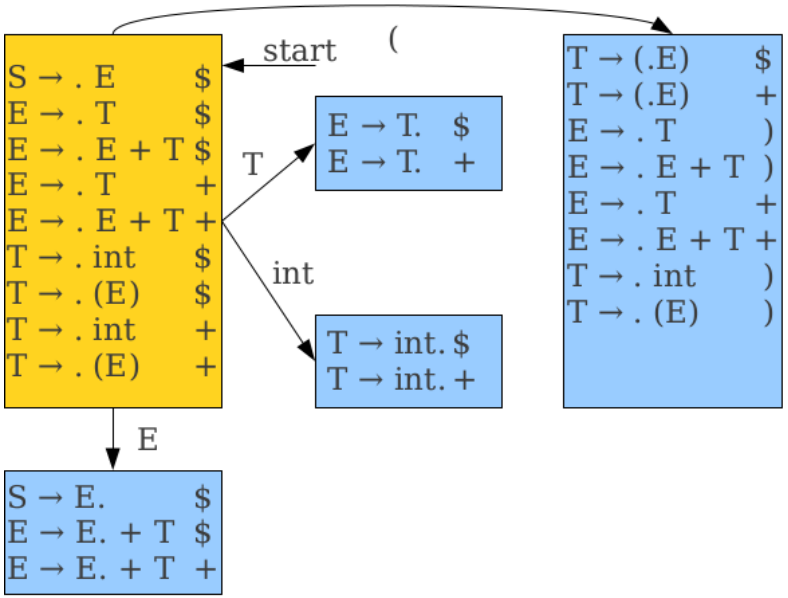
Deterministic LR(1) Automata



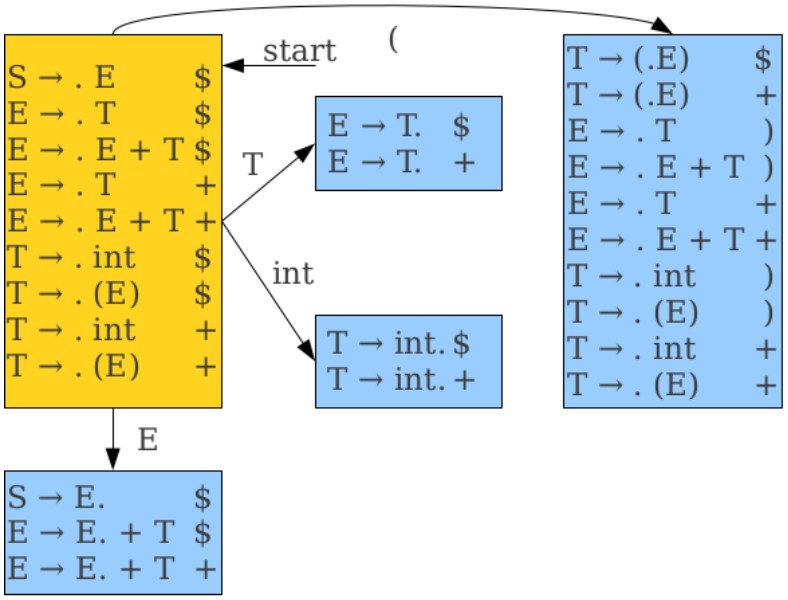
Deterministic LR(1) Automata



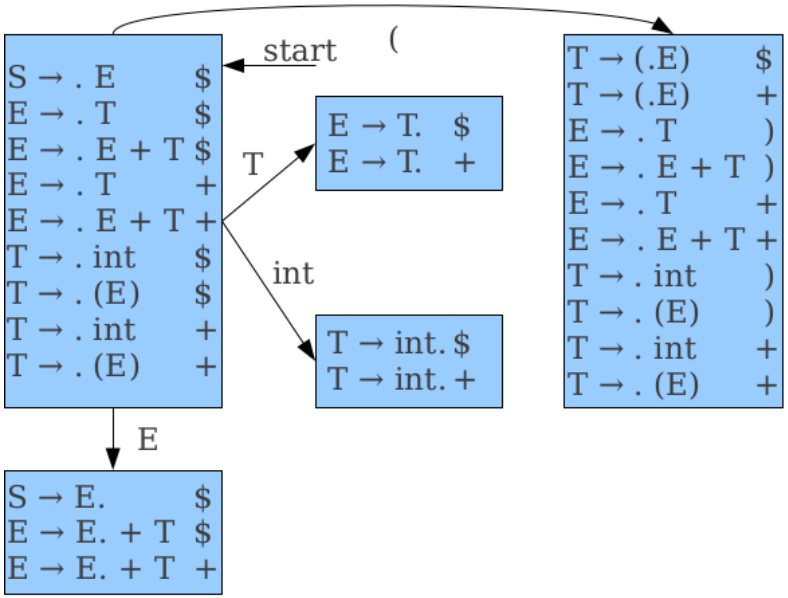
Deterministic LR(1) Automata



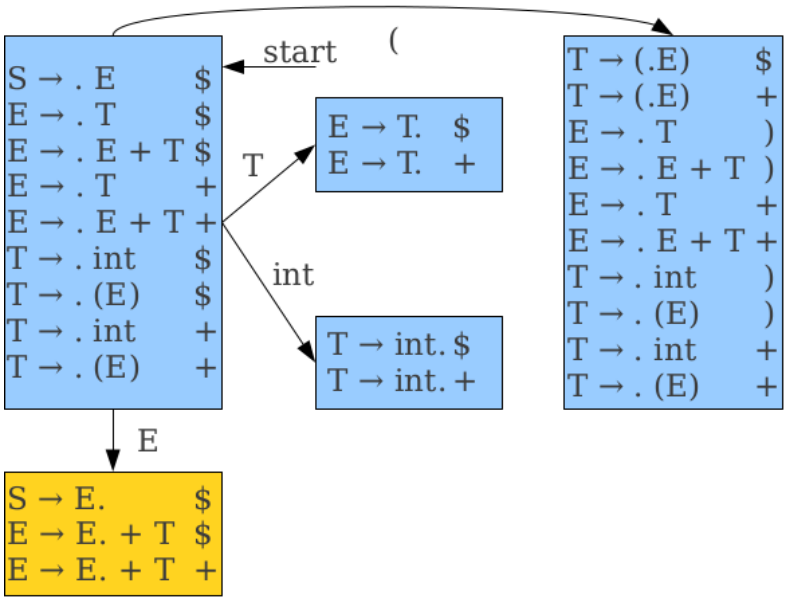
Deterministic LR(1) Automata



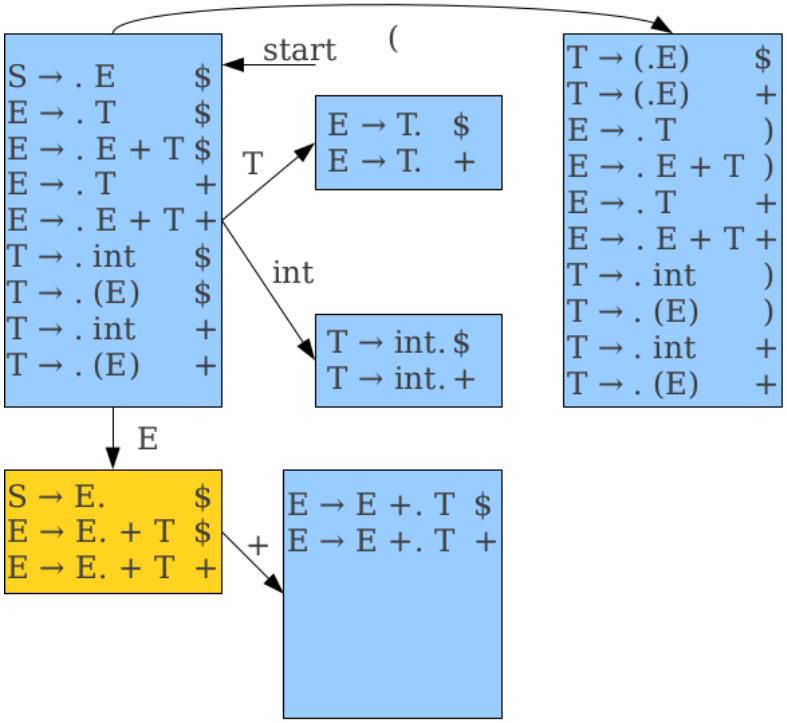
Deterministic LR(1) Automata



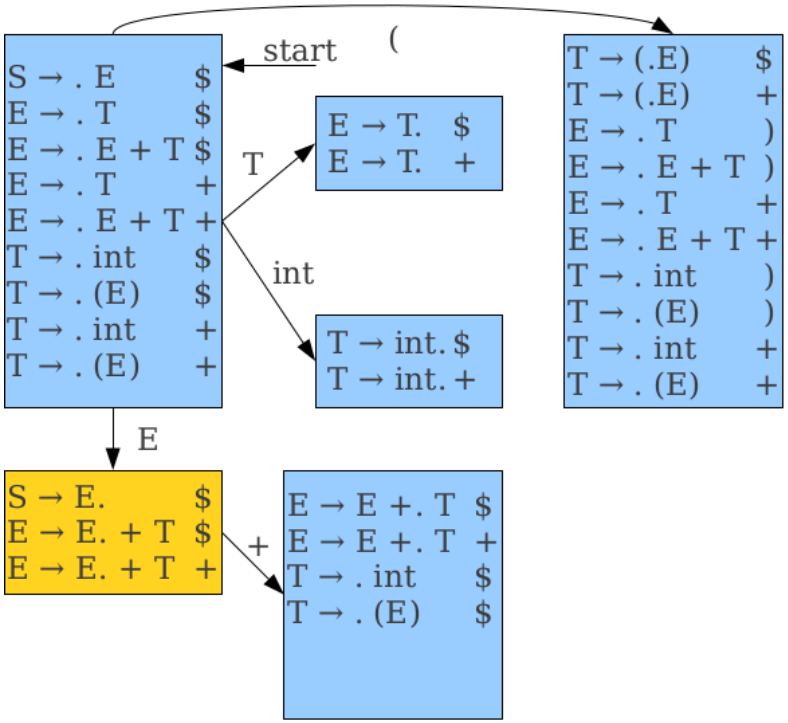
Deterministic LR(1) Automata



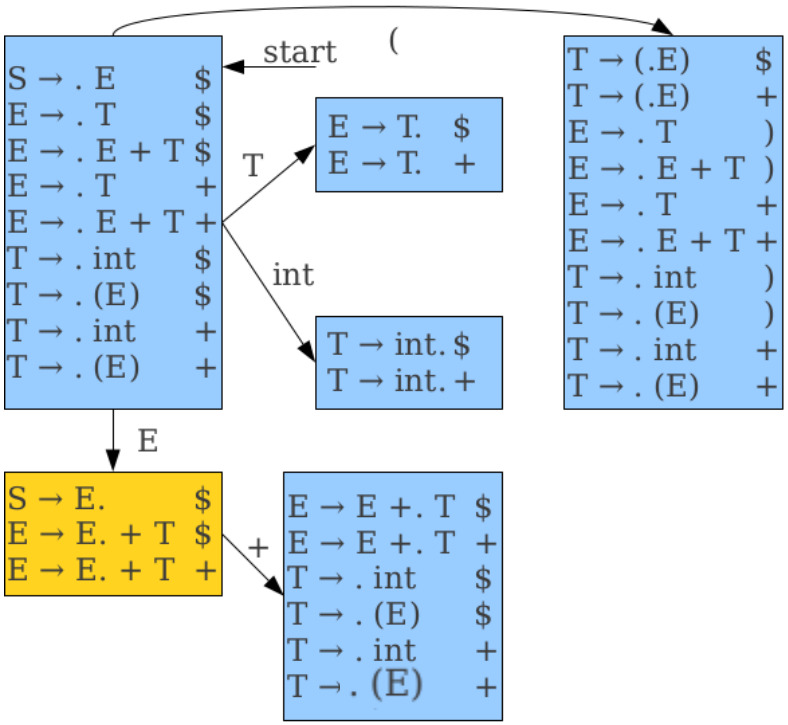
Deterministic LR(1) Automata



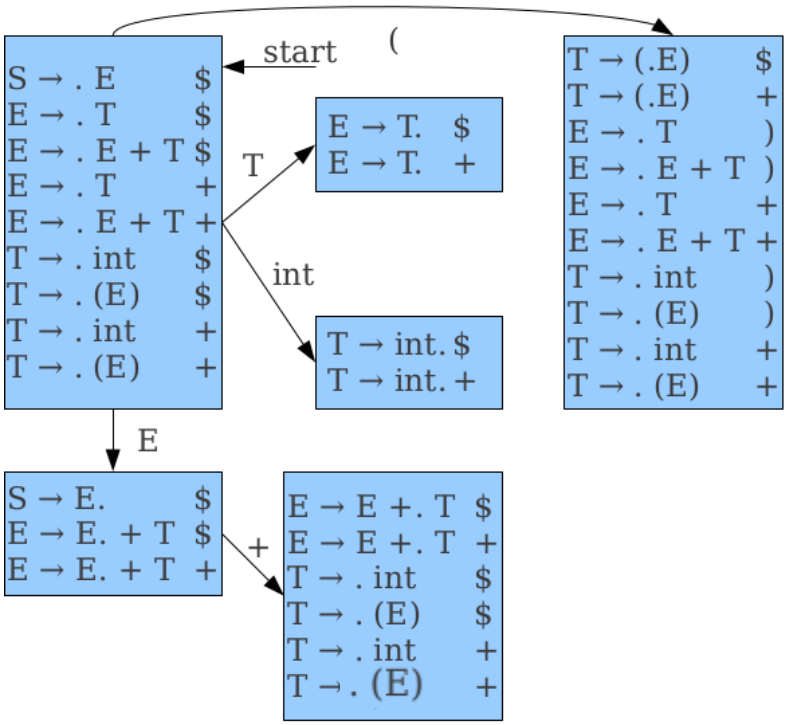
Deterministic LR(1) Automata



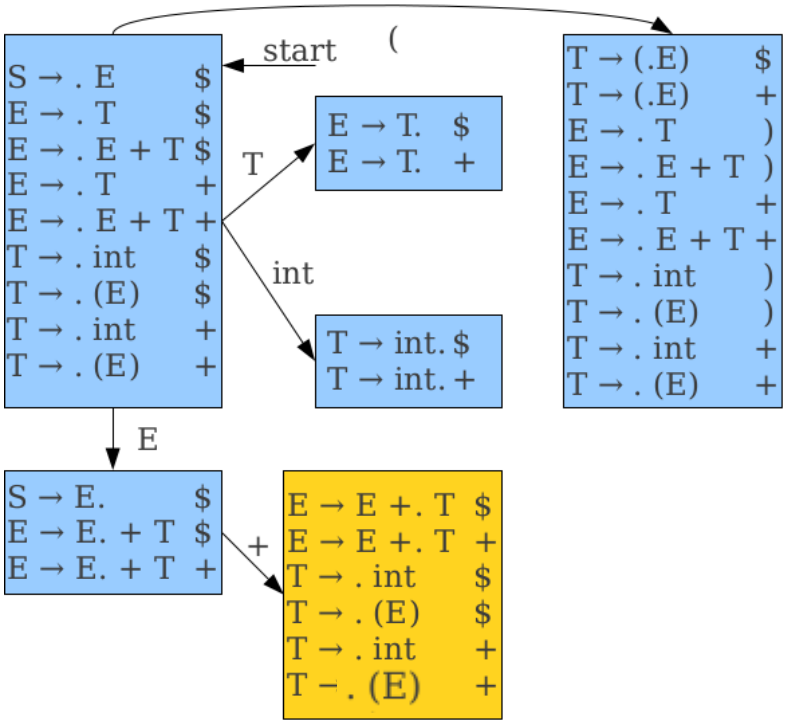
Deterministic LR(1) Automata



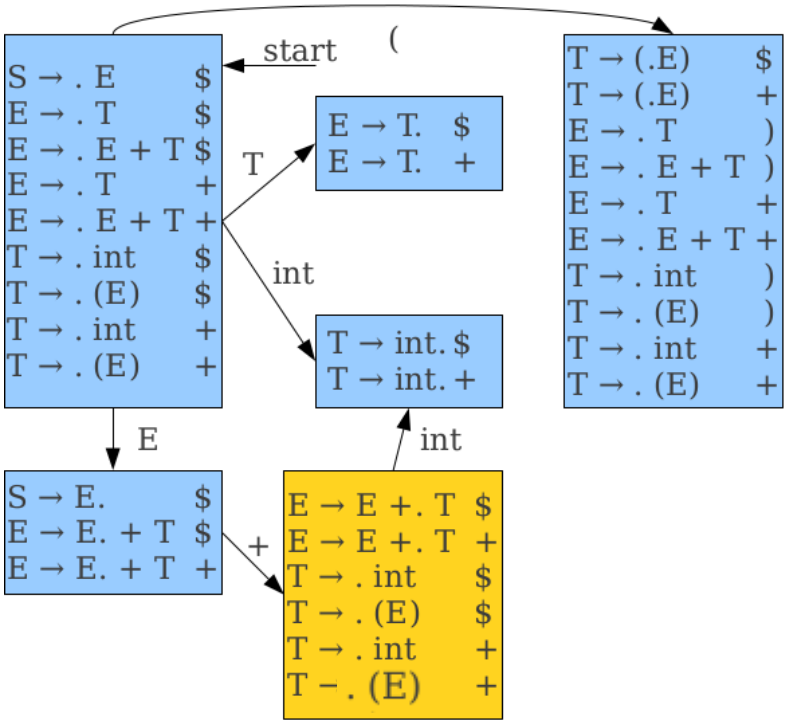
Deterministic LR(1) Automata



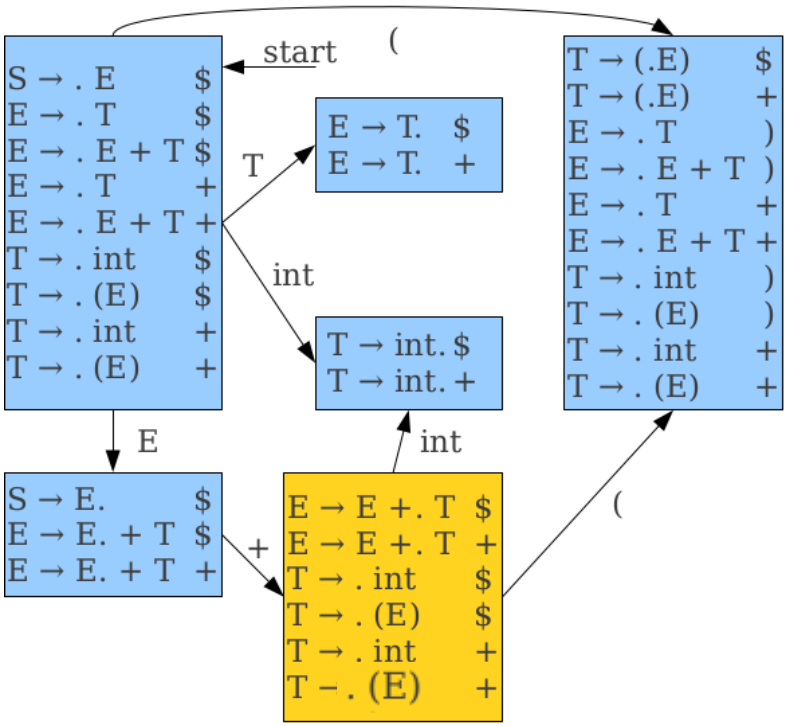
Deterministic LR(1) Automata



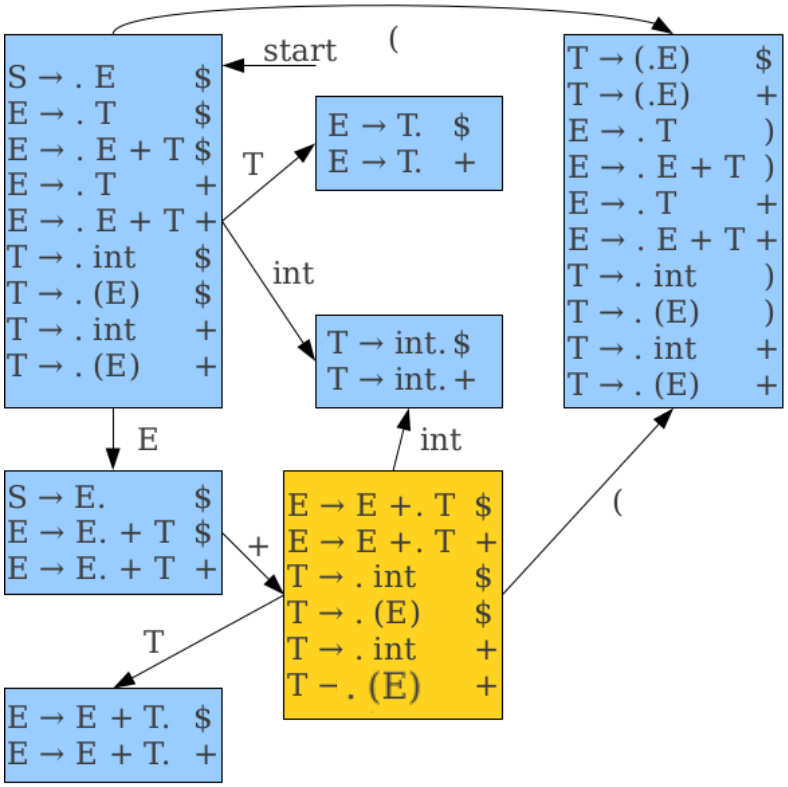
Deterministic LR(1) Automata



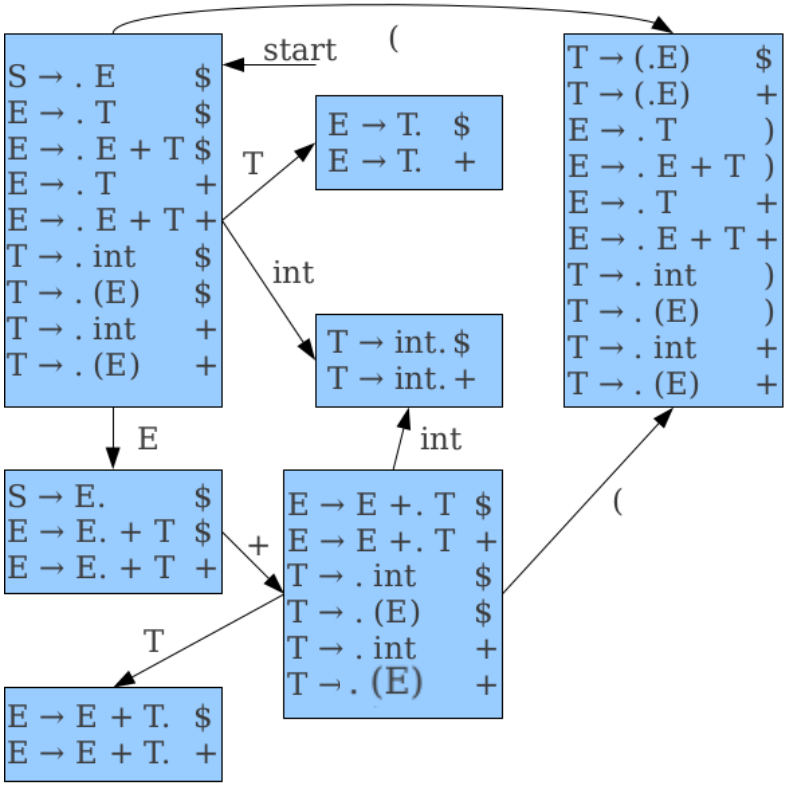
Deterministic LR(1) Automata



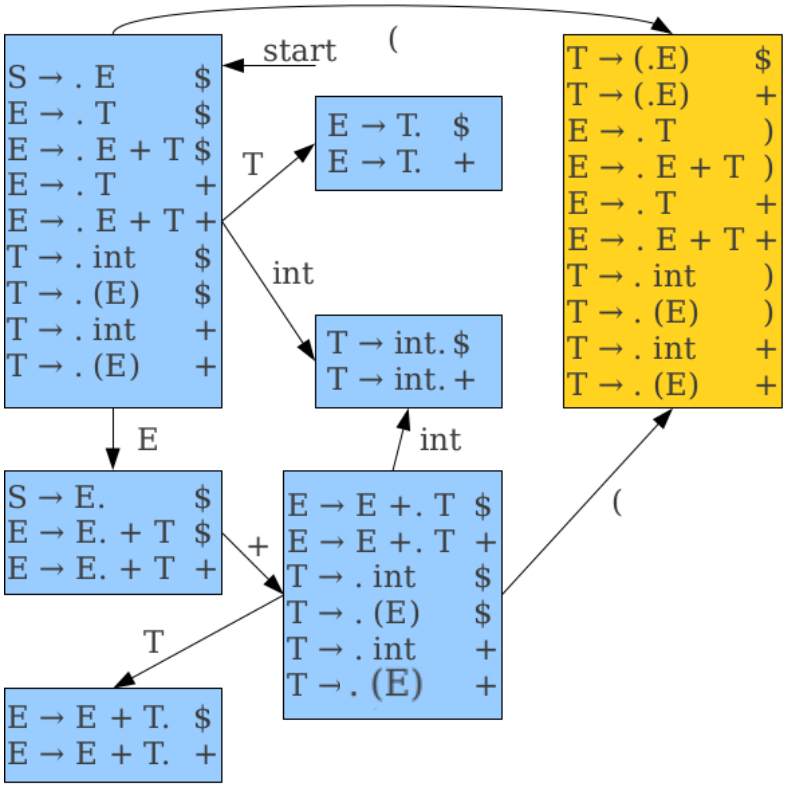
Deterministic LR(1) Automata



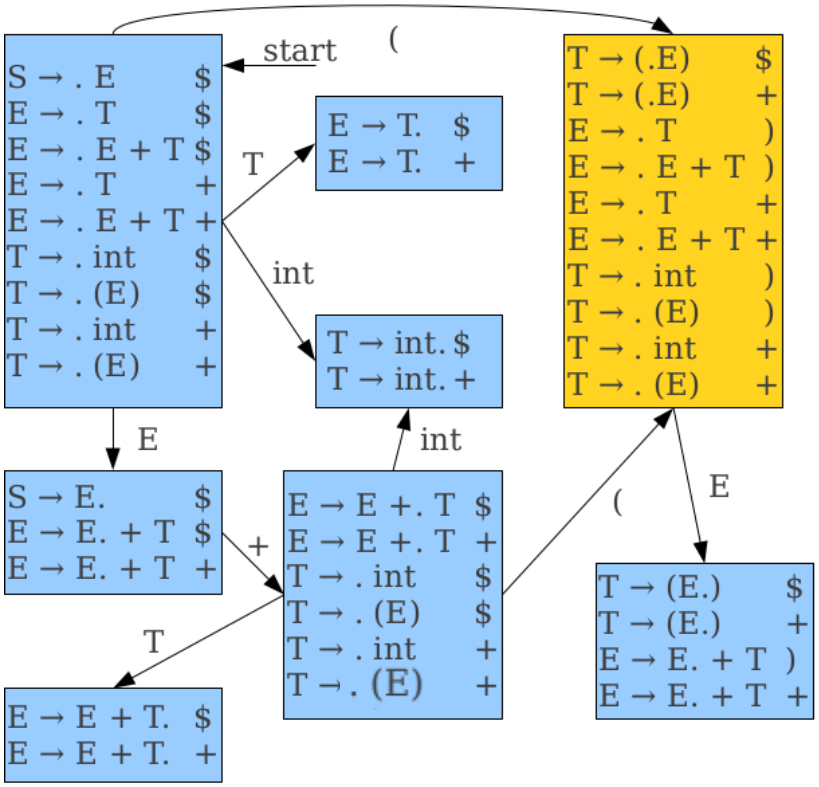
Deterministic LR(1) Automata



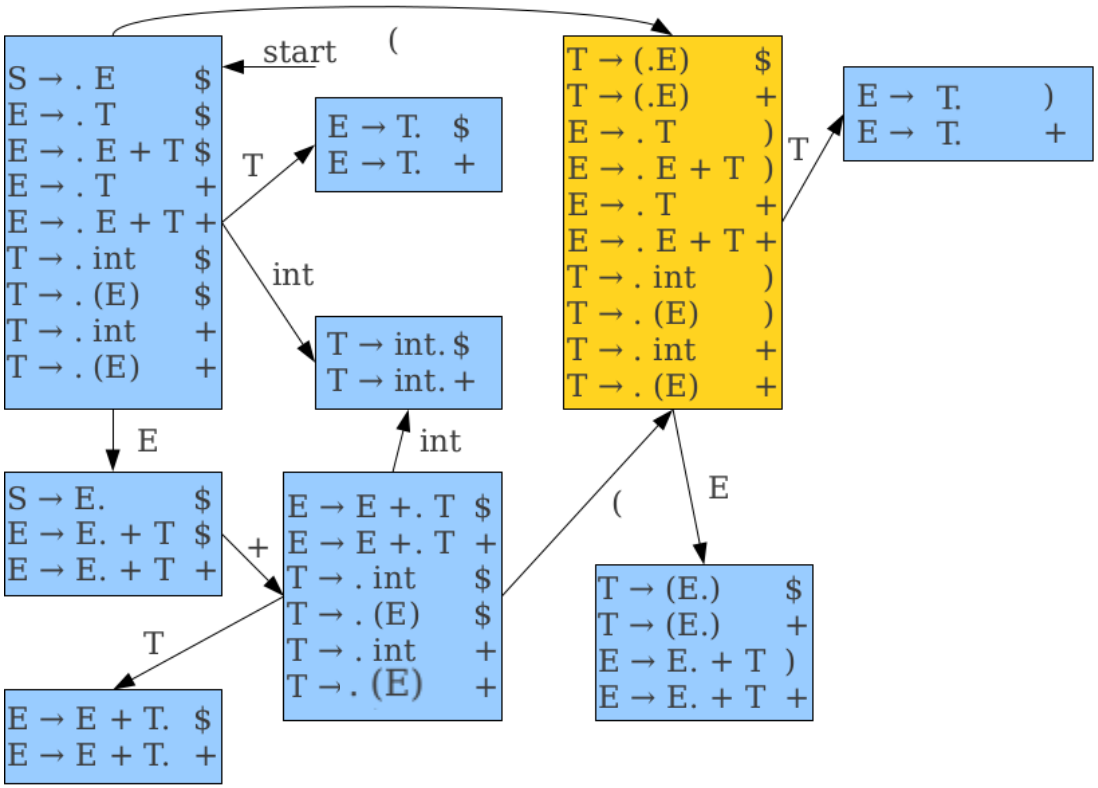
Deterministic LR(1) Automata



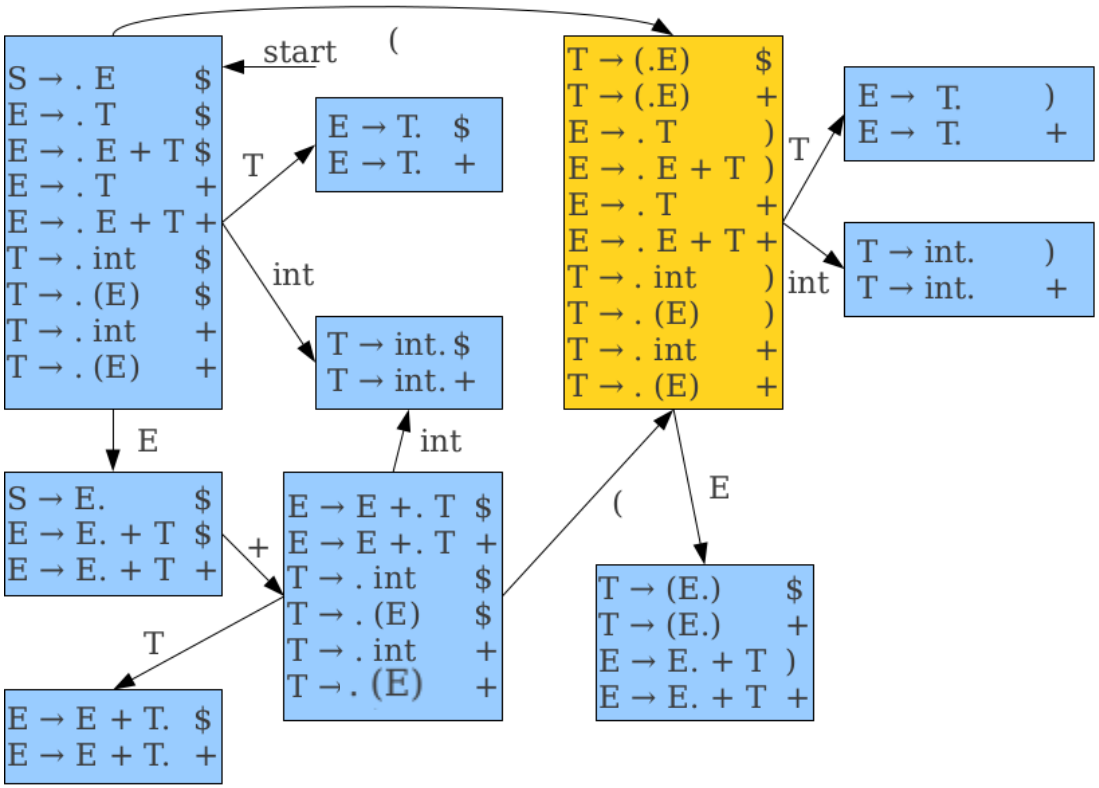
Deterministic LR(1) Automata



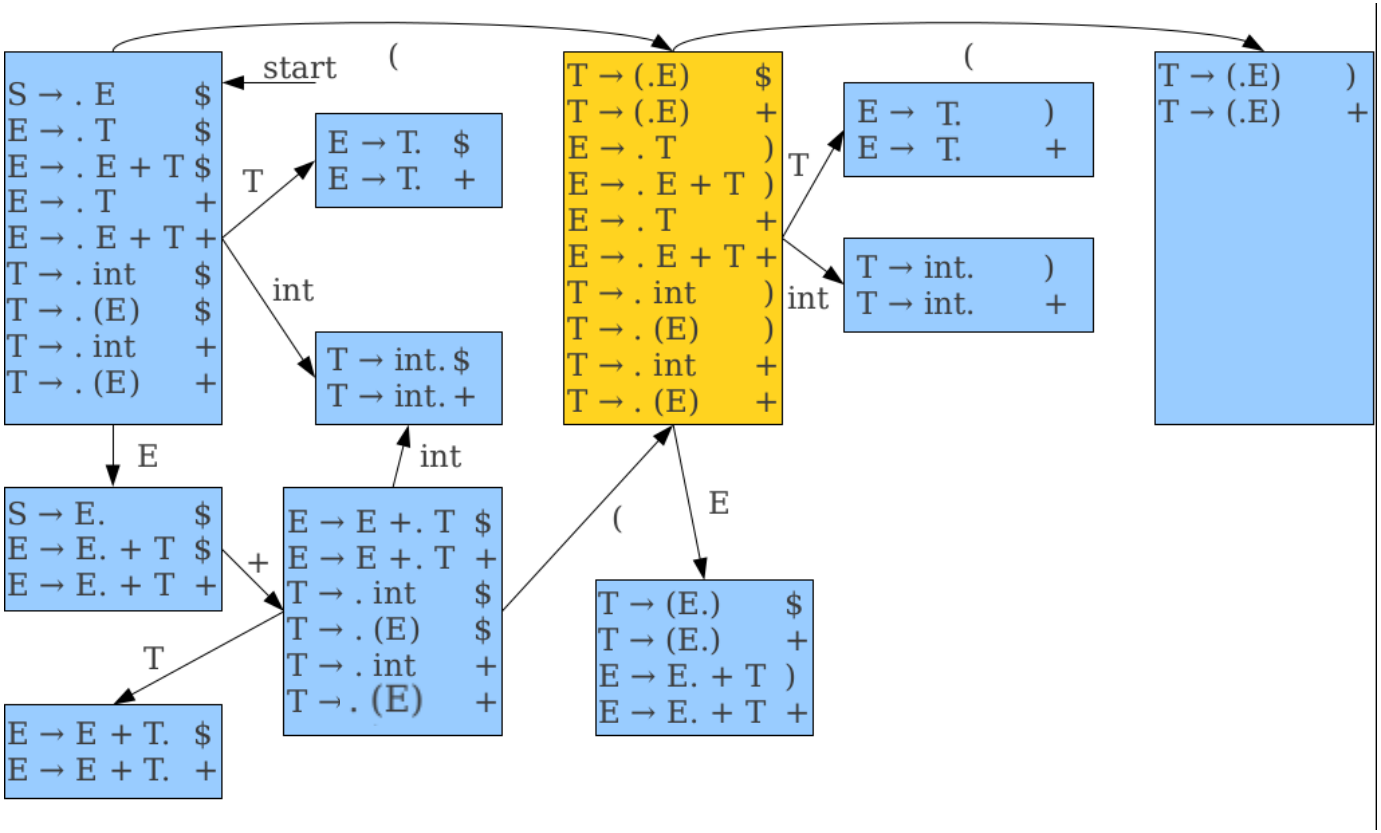
Deterministic LR(1) Automata



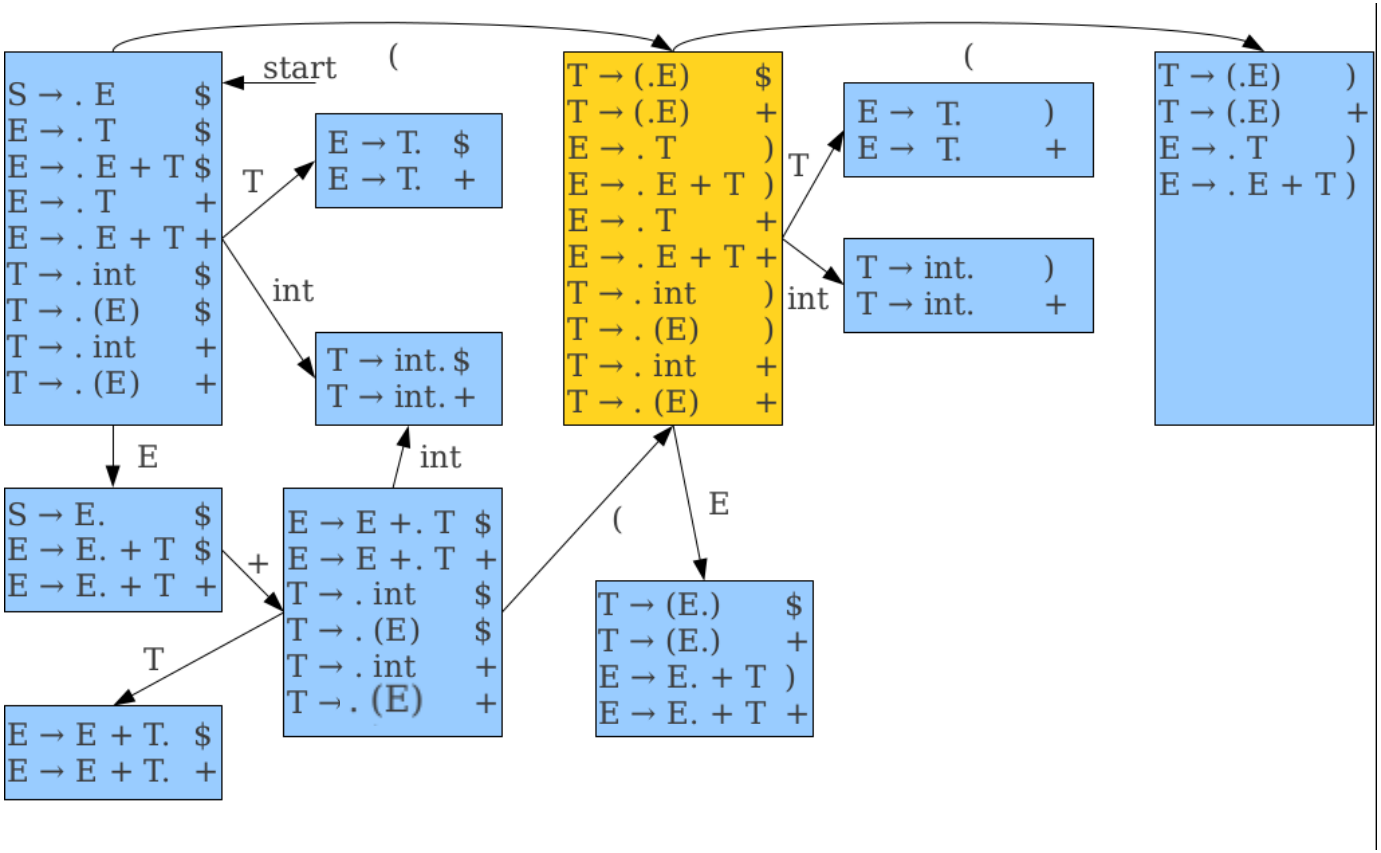
Deterministic LR(1) Automata



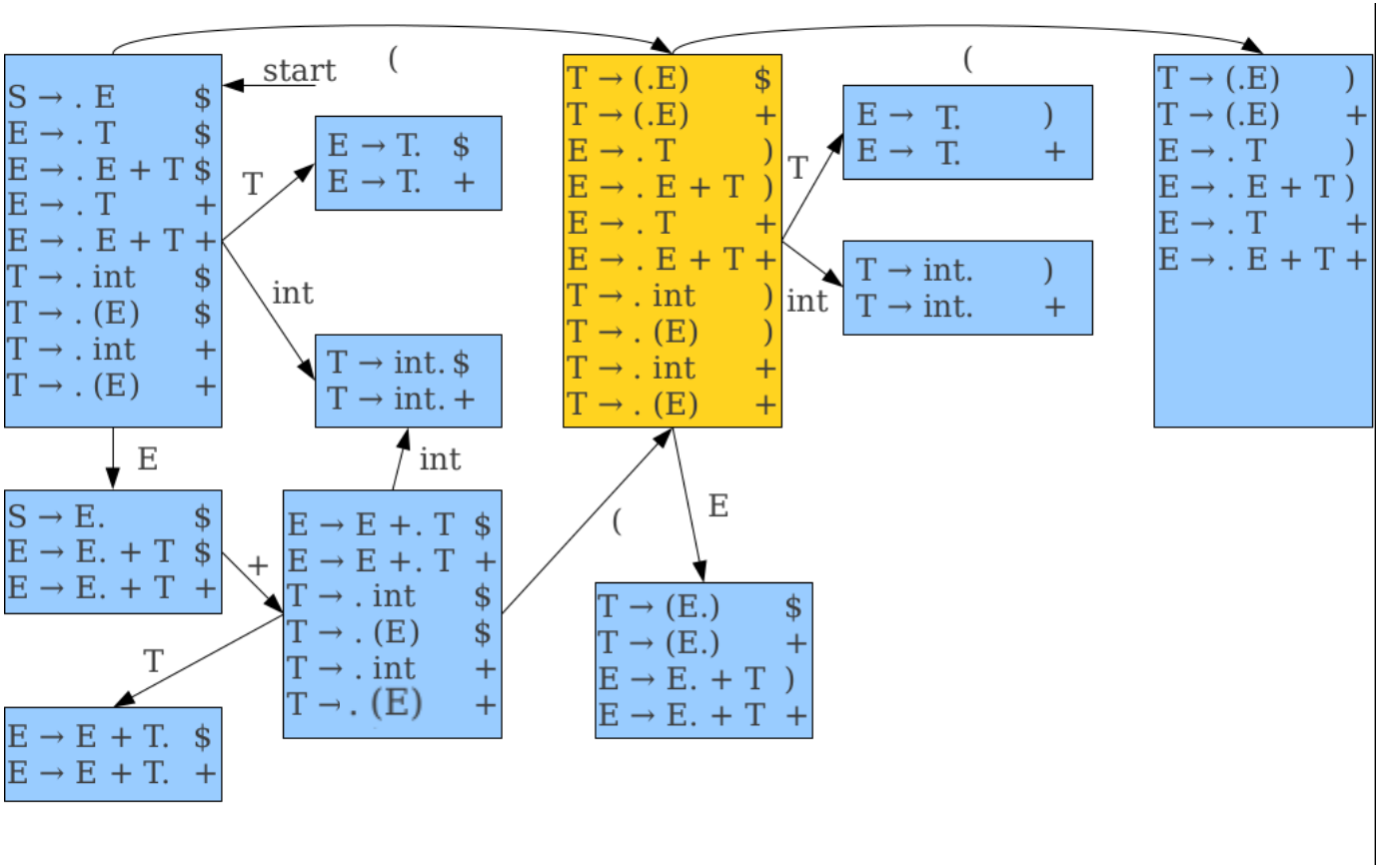
Deterministic LR(1) Automata



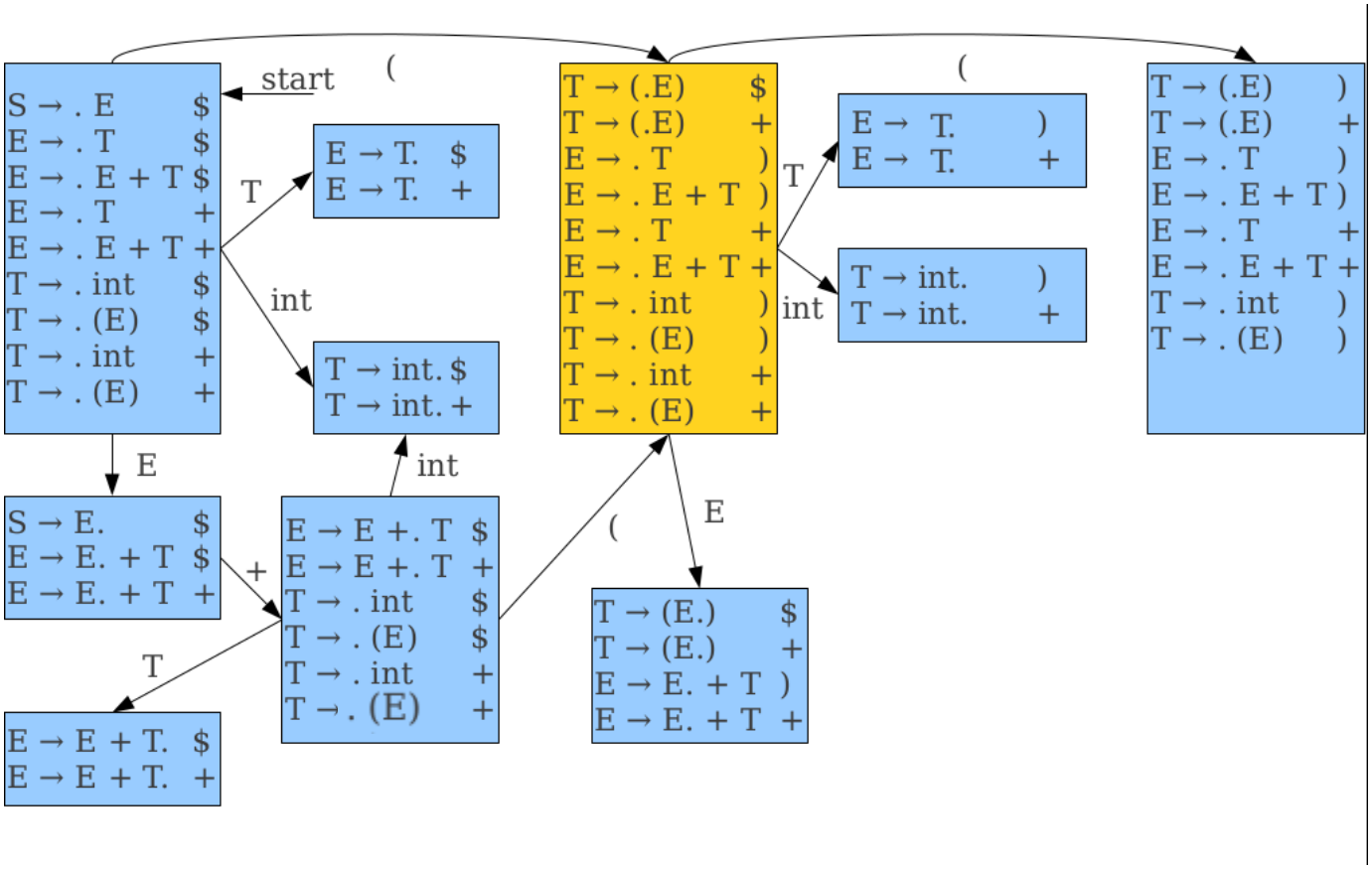
Deterministic LR(1) Automata



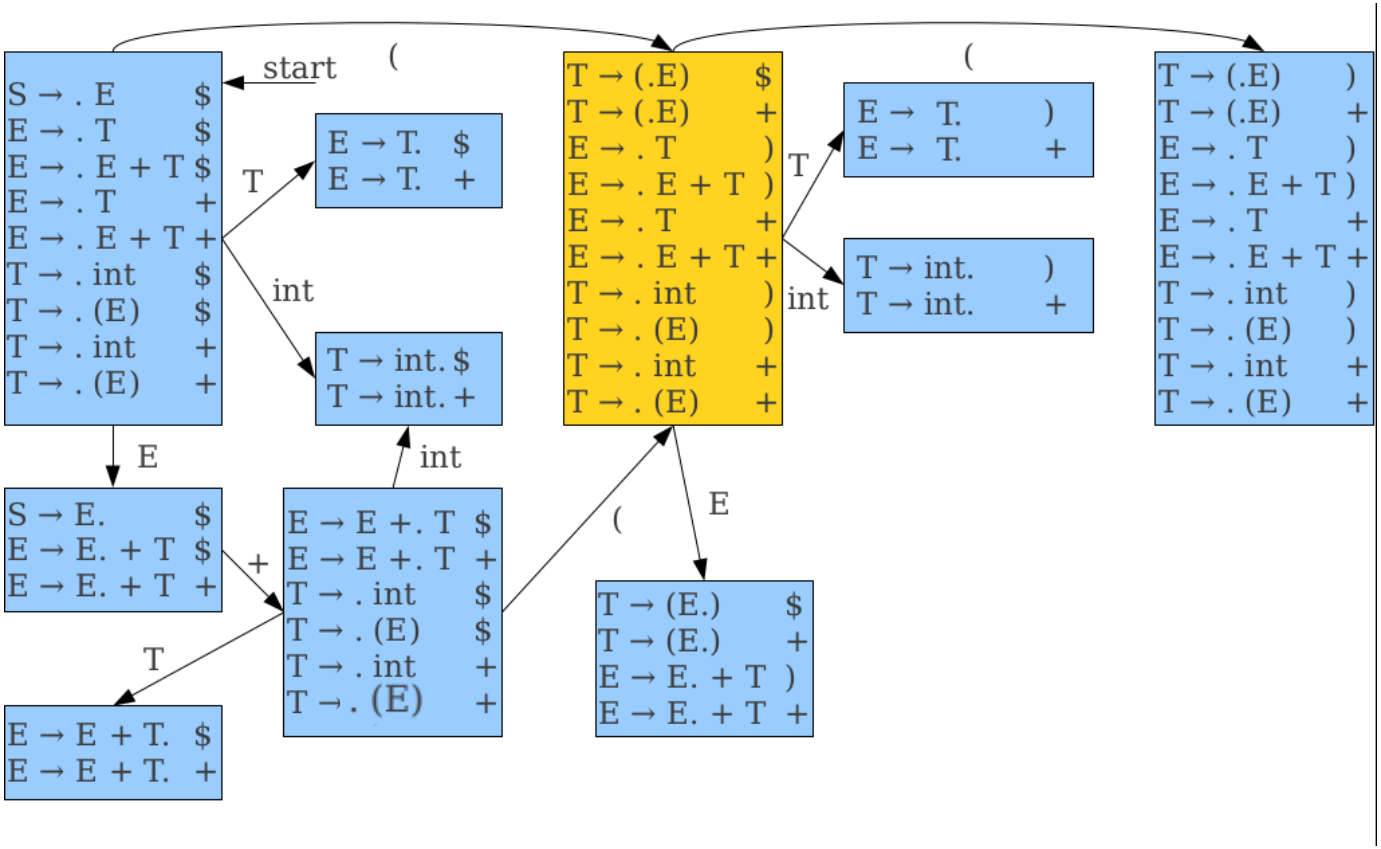
Deterministic LR(1) Automata



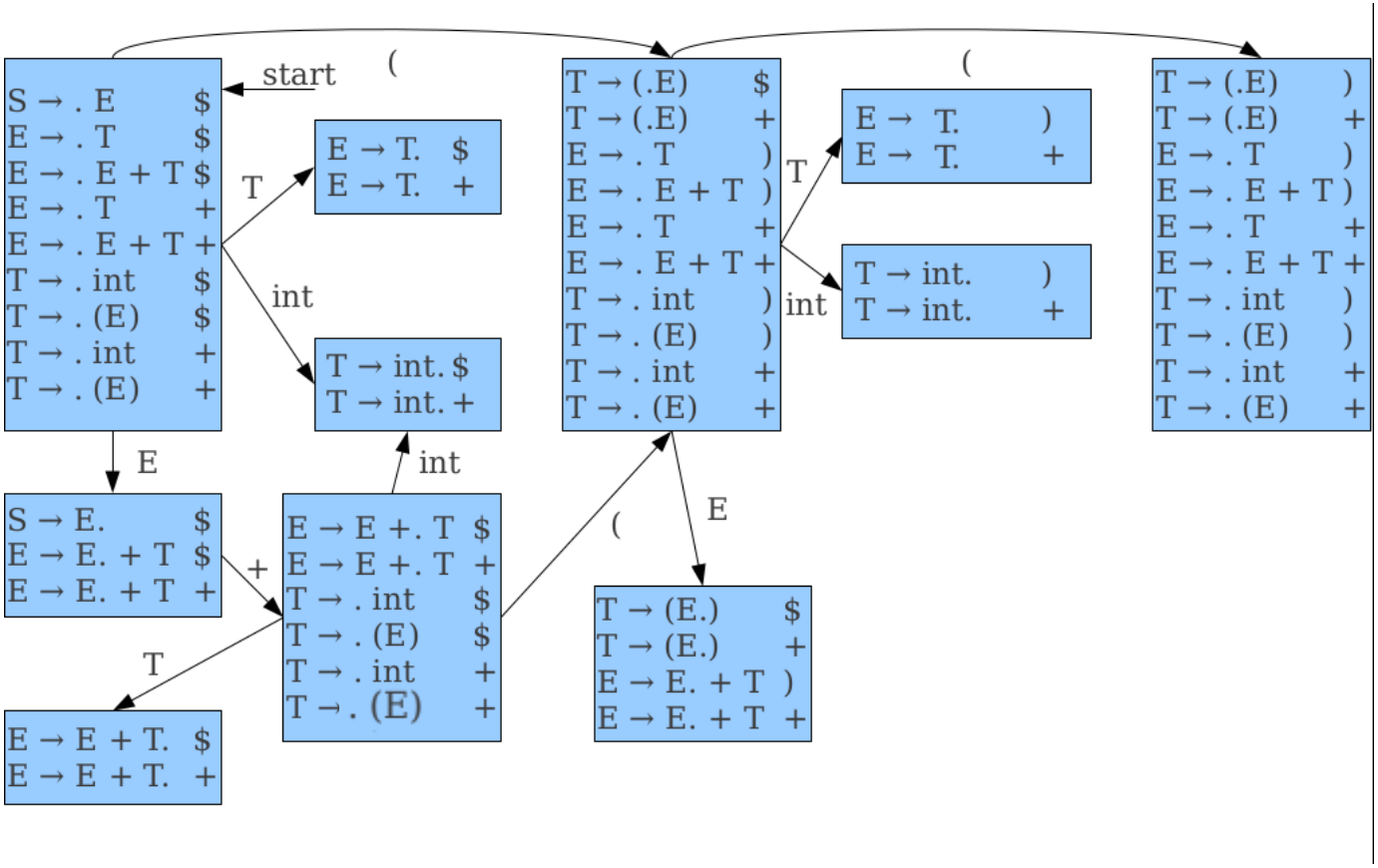
Deterministic LR(1) Automata



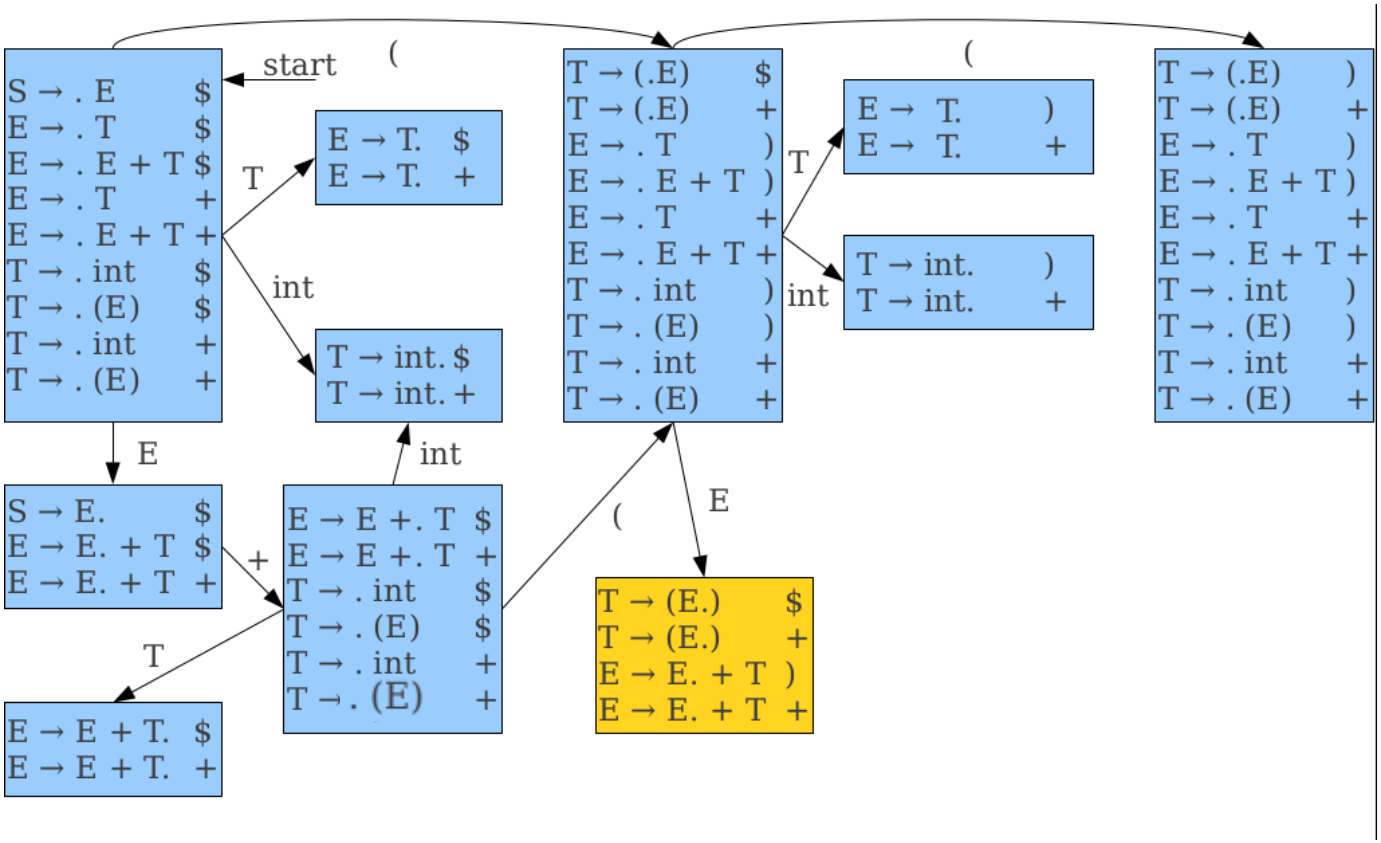
Deterministic LR(1) Automata



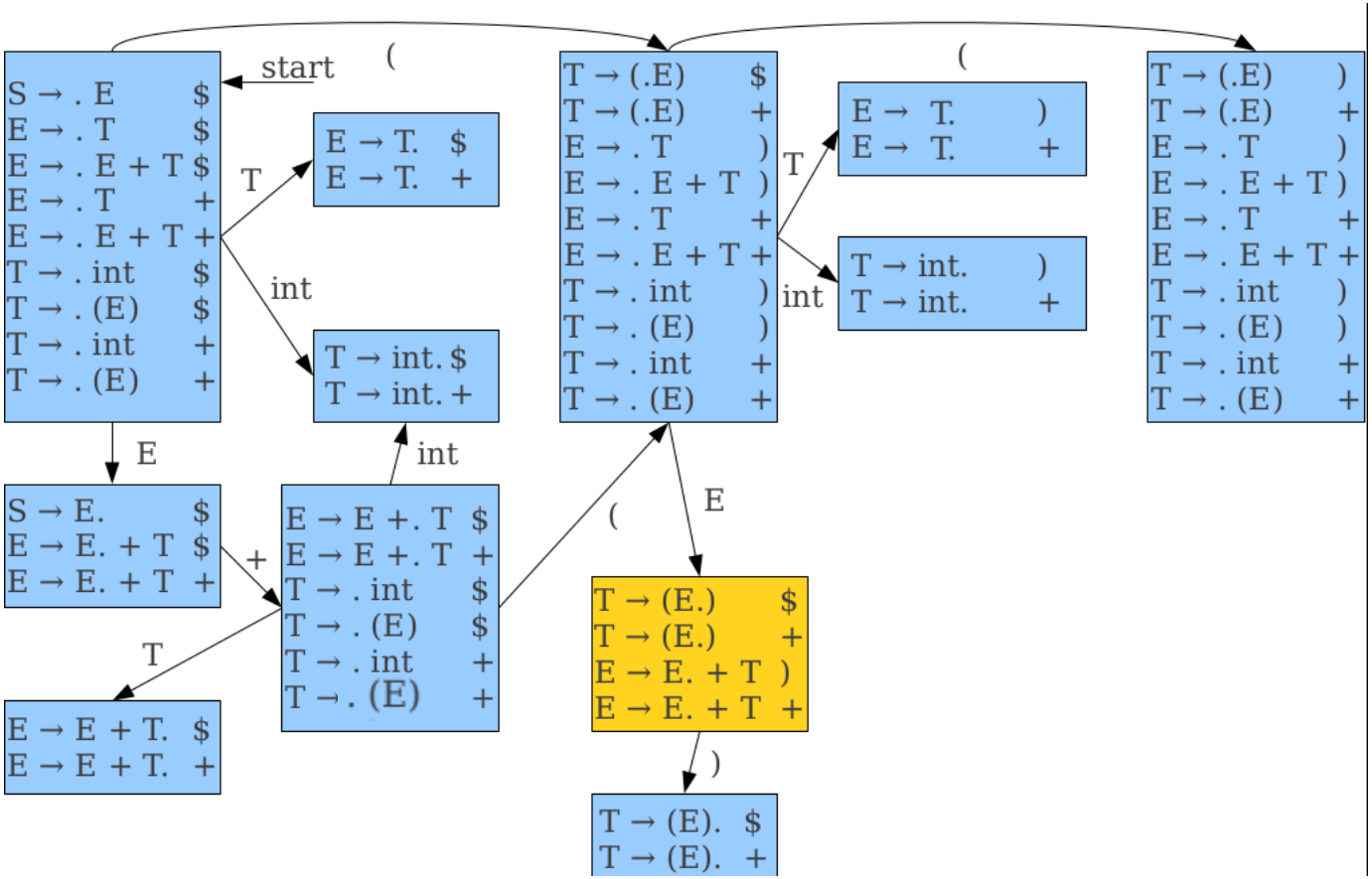
Deterministic LR(1) Automata



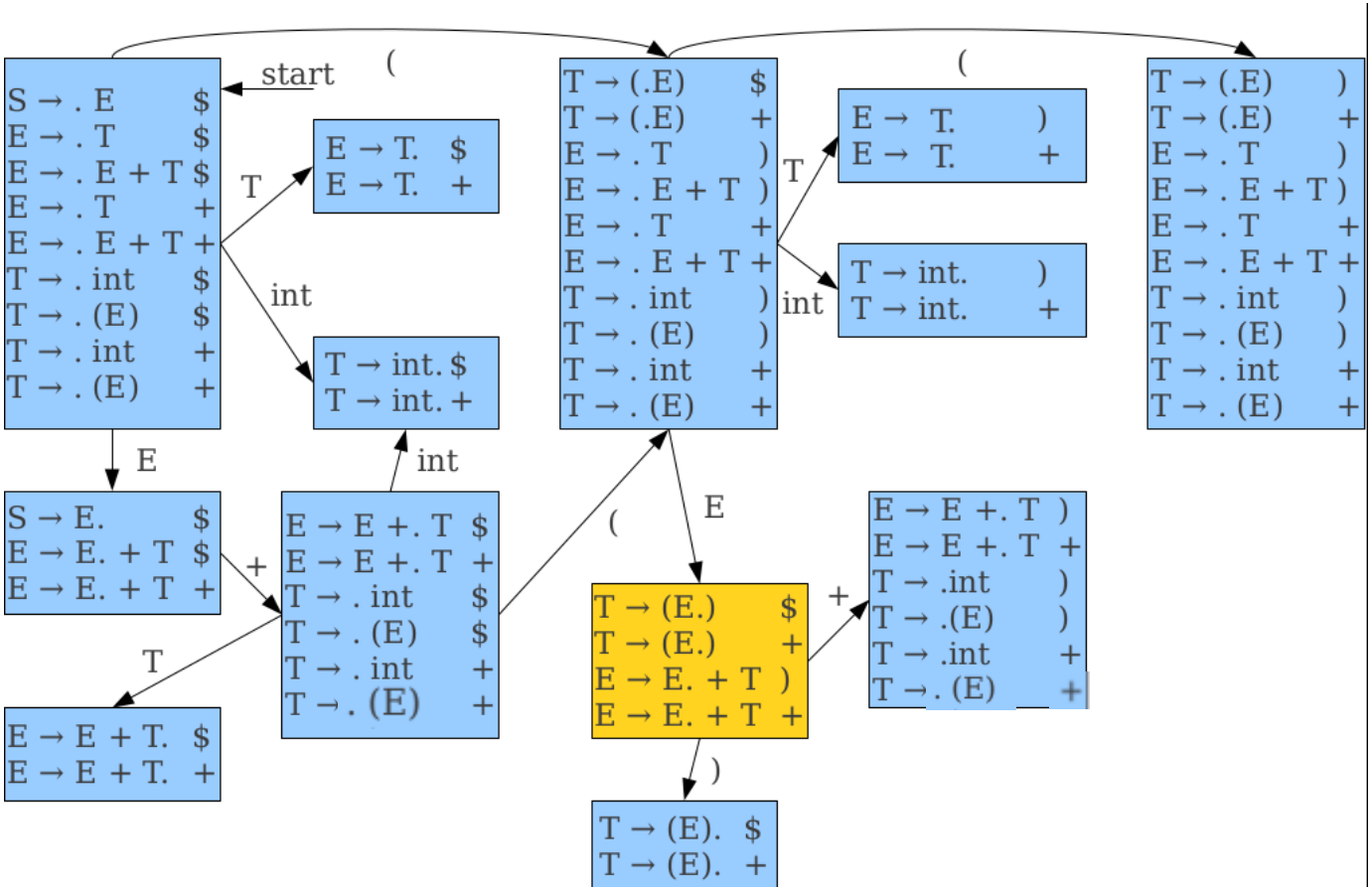
Deterministic LR(1) Automata



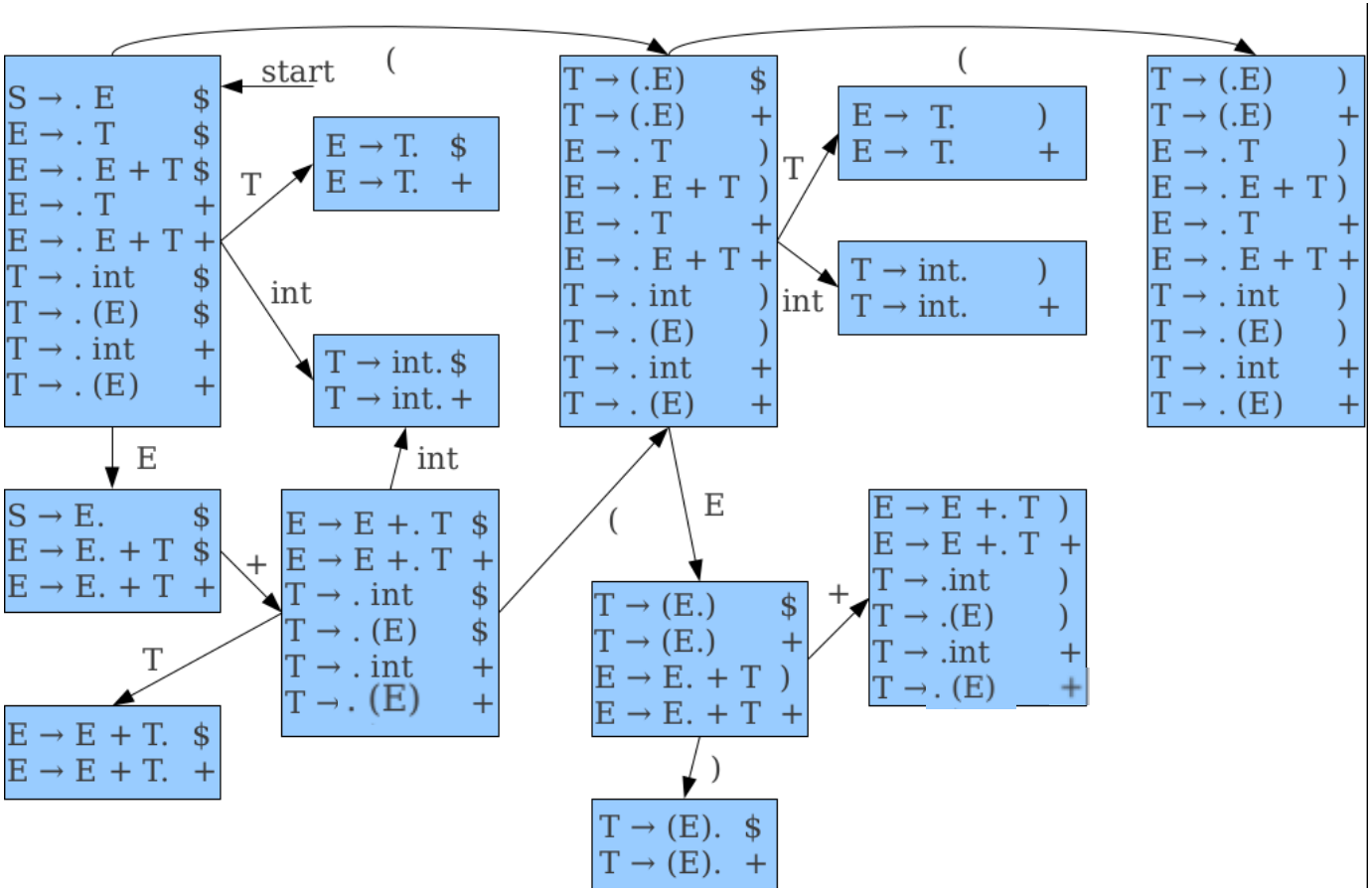
Deterministic LR(1) Automata



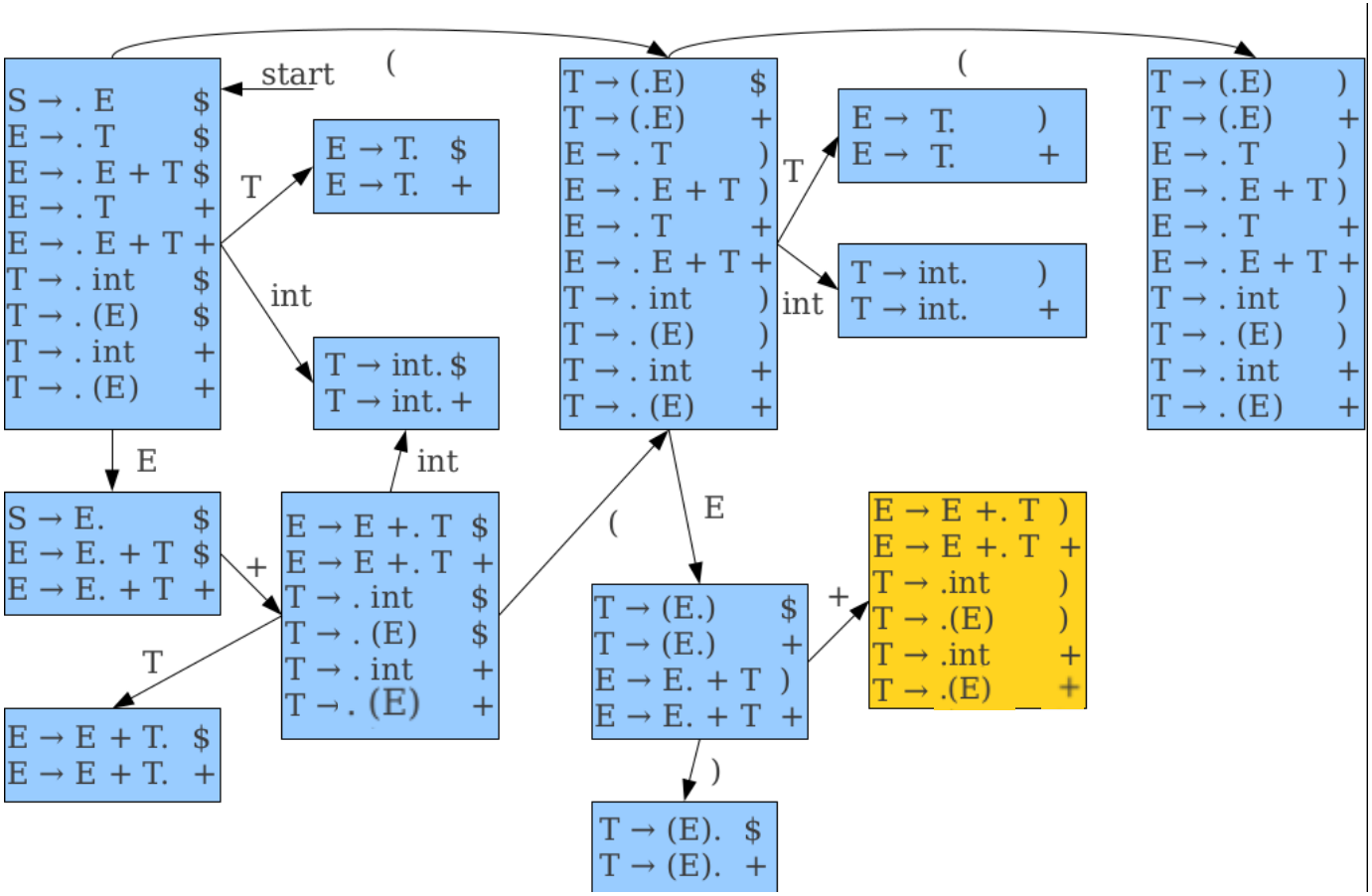
Deterministic LR(1) Automata



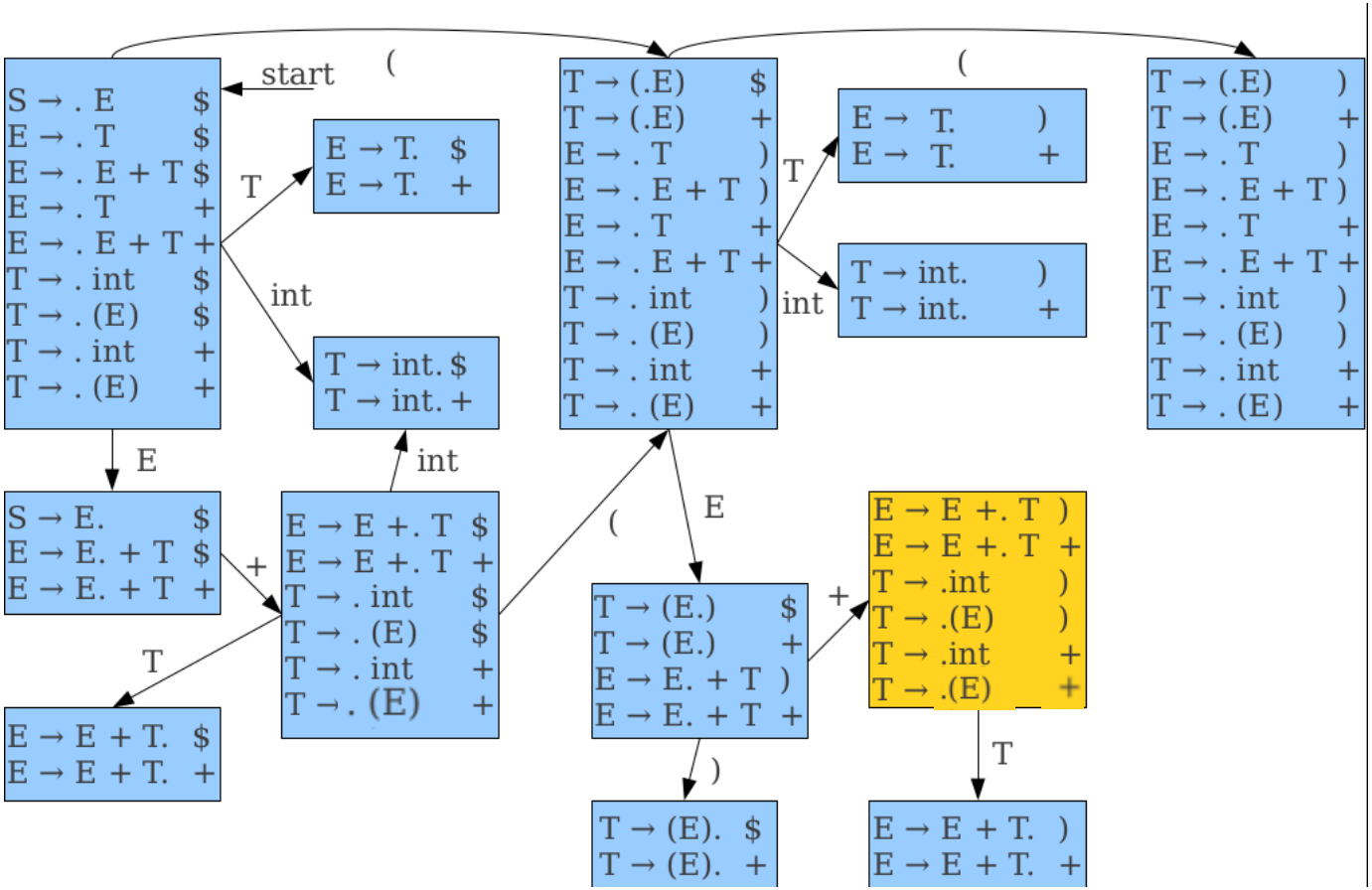
Deterministic LR(1) Automata



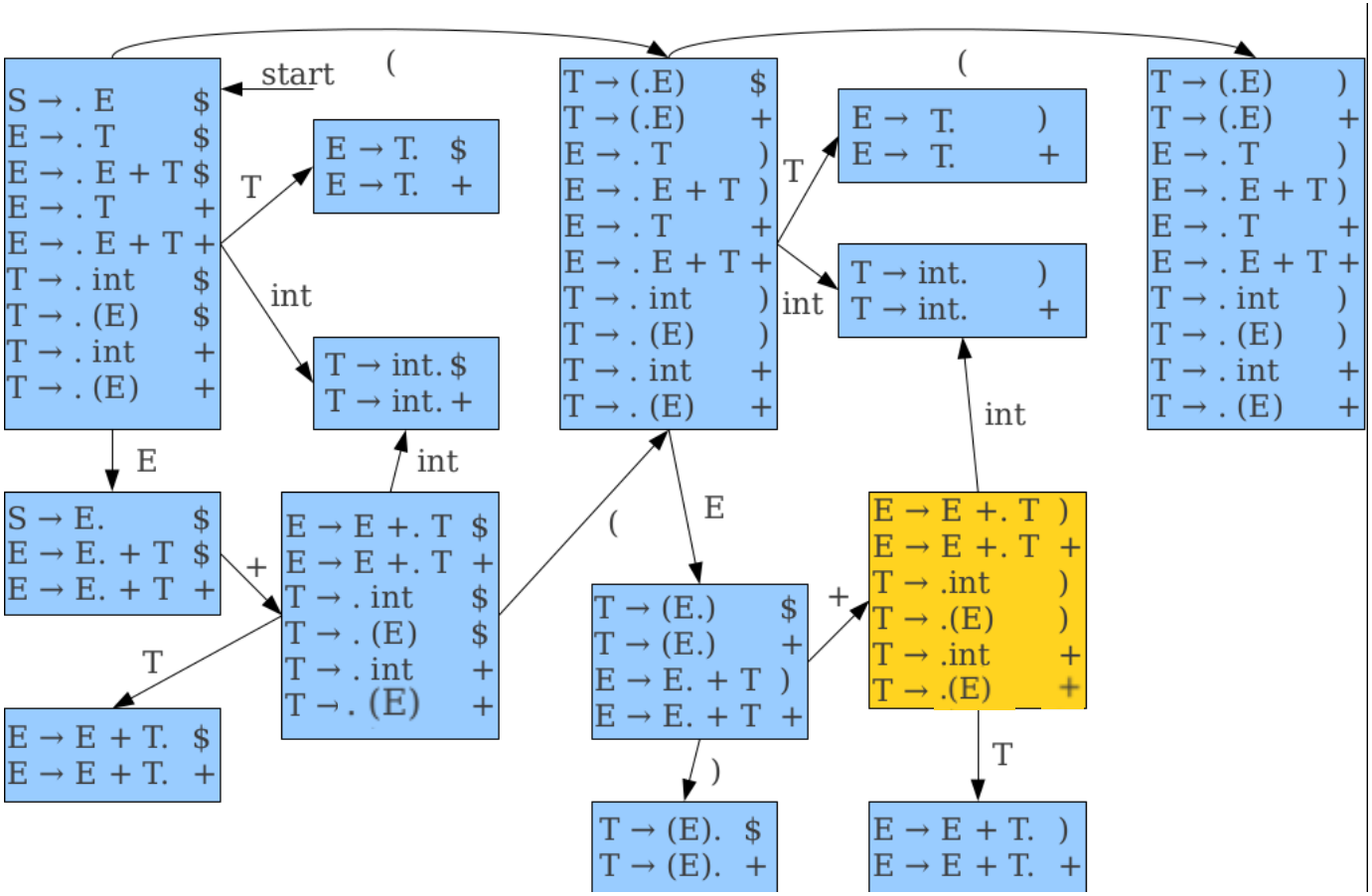
Deterministic LR(1) Automata



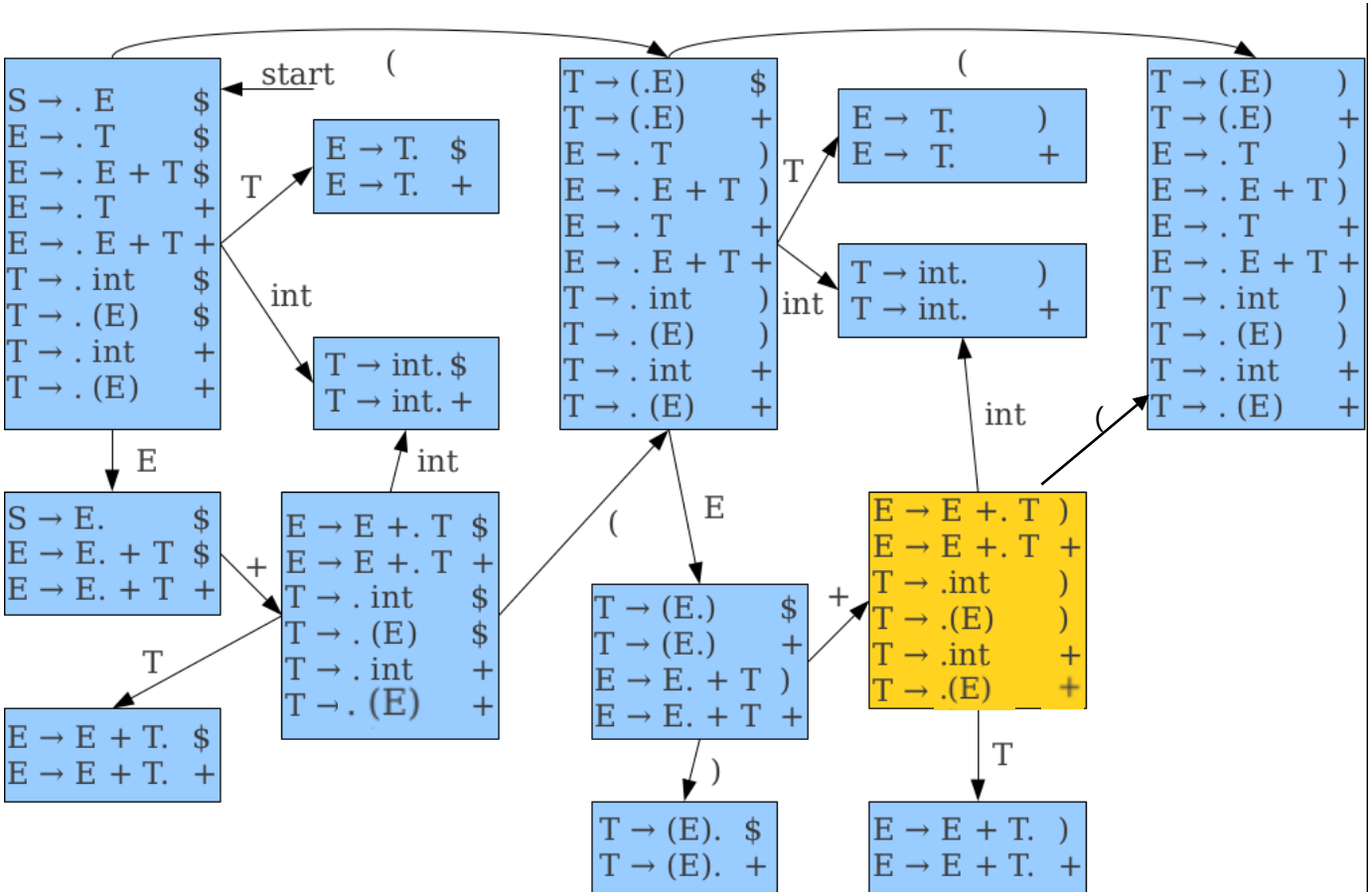
Deterministic LR(1) Automata



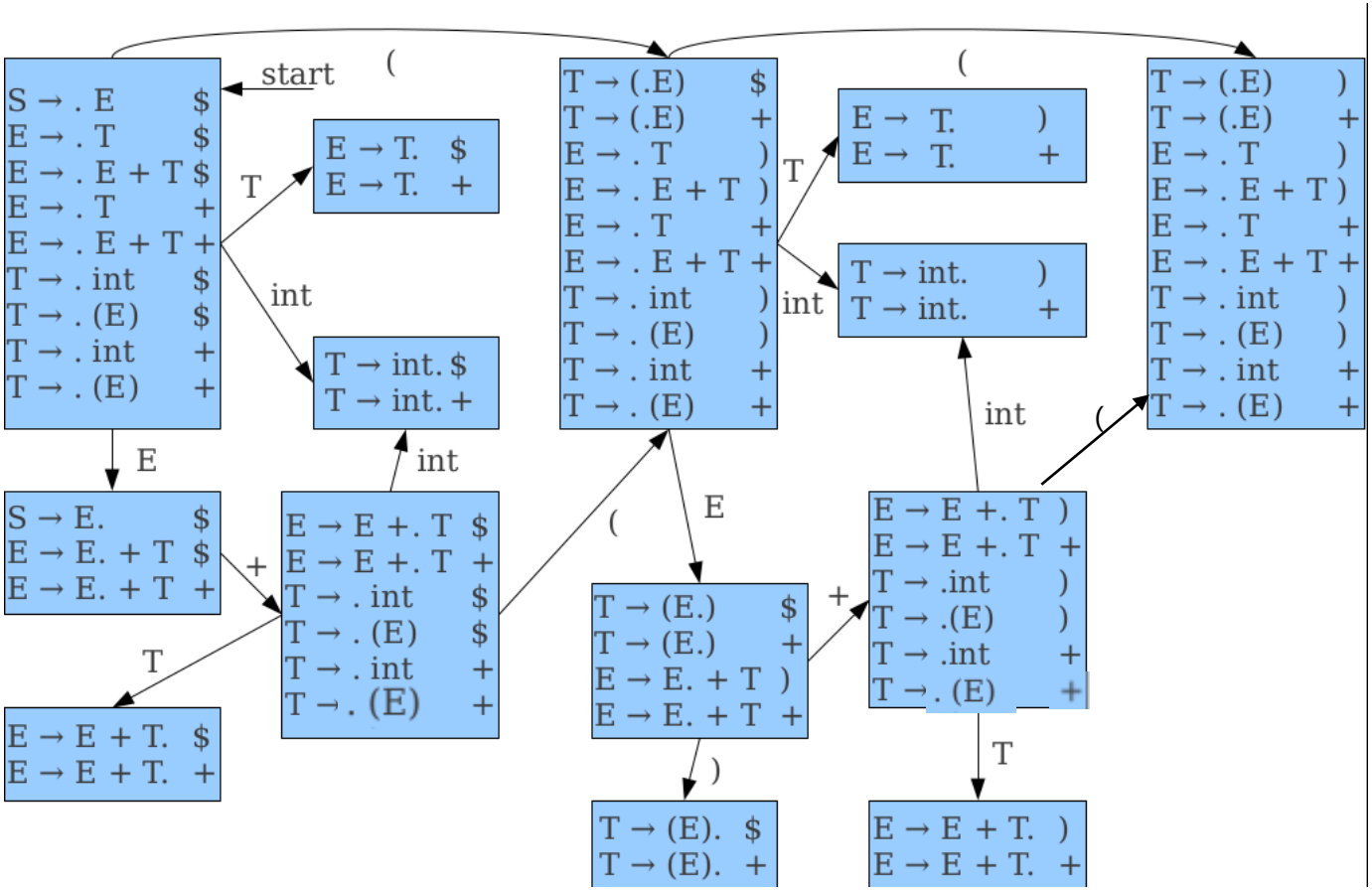
Deterministic LR(1) Automata



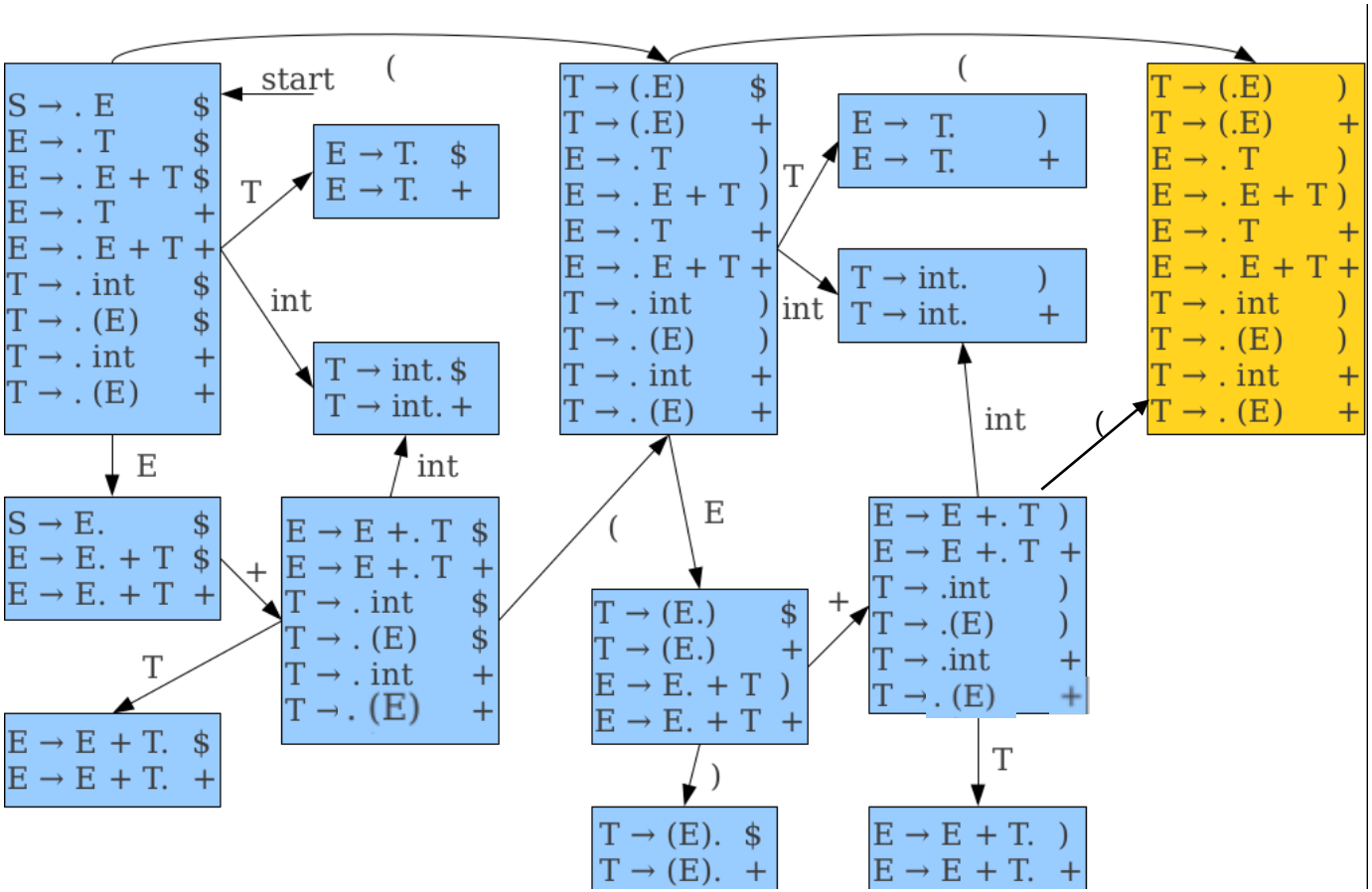
Deterministic LR(1) Automata



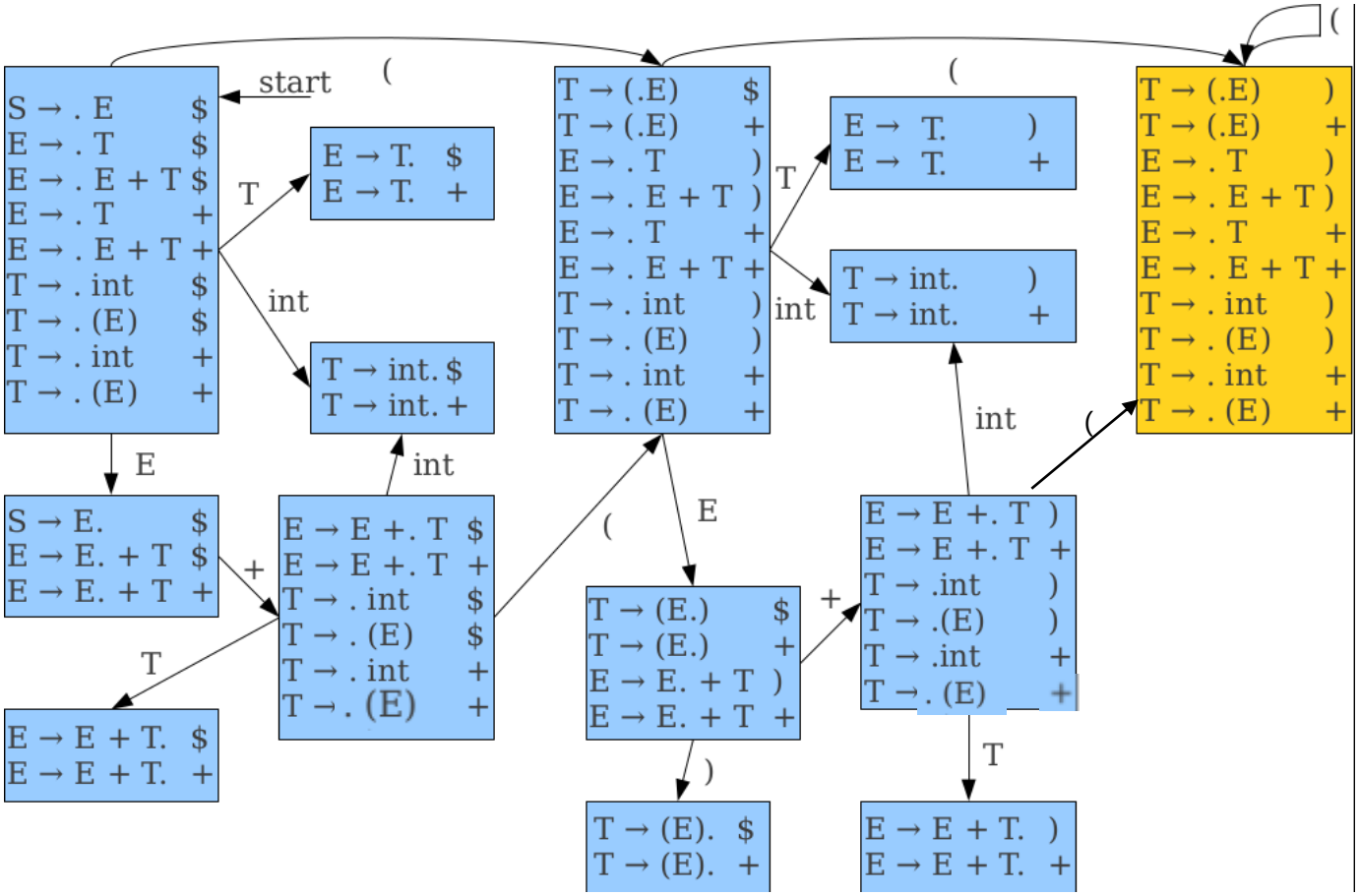
Deterministic LR(1) Automata



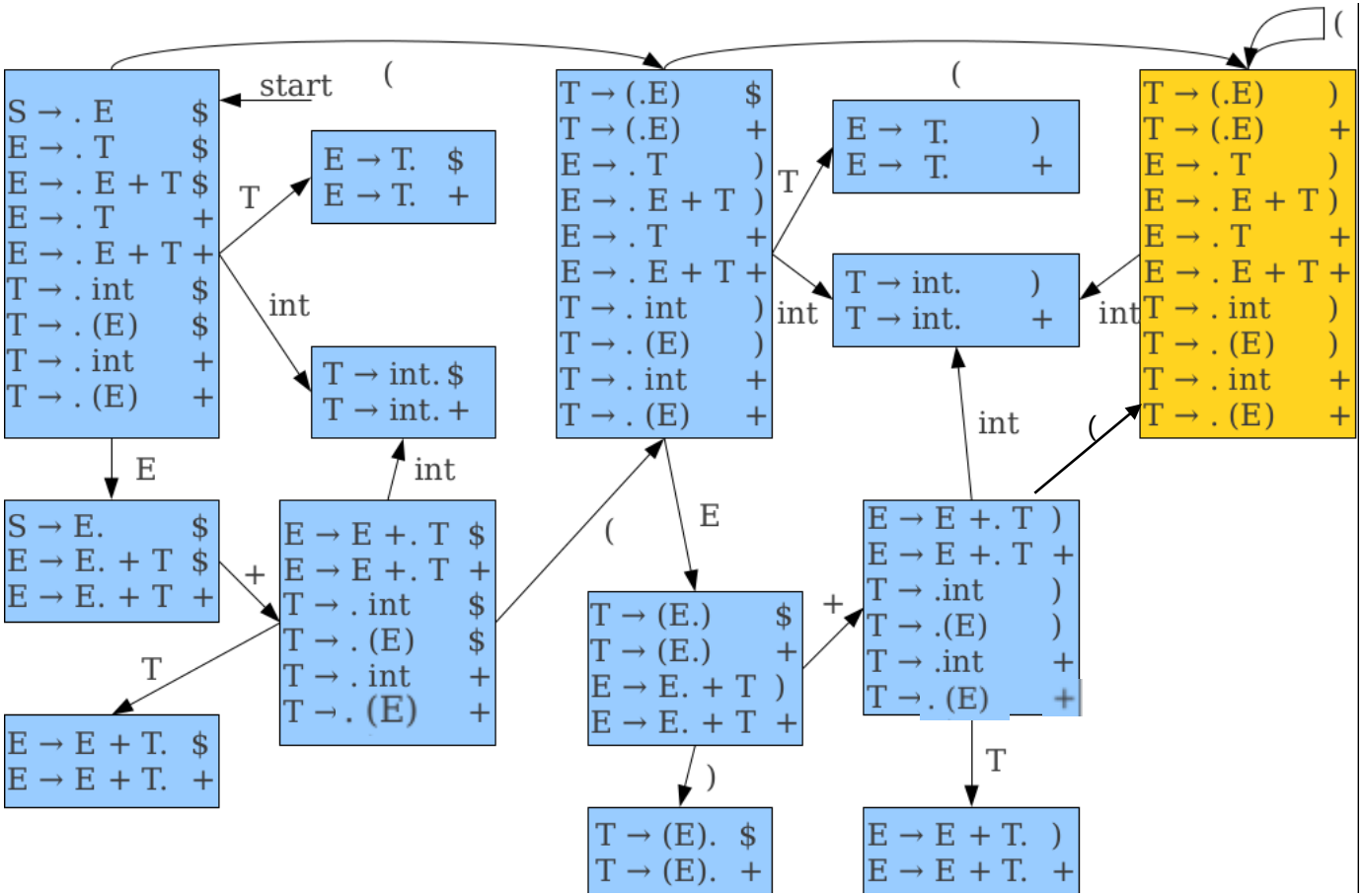
Deterministic LR(1) Automata



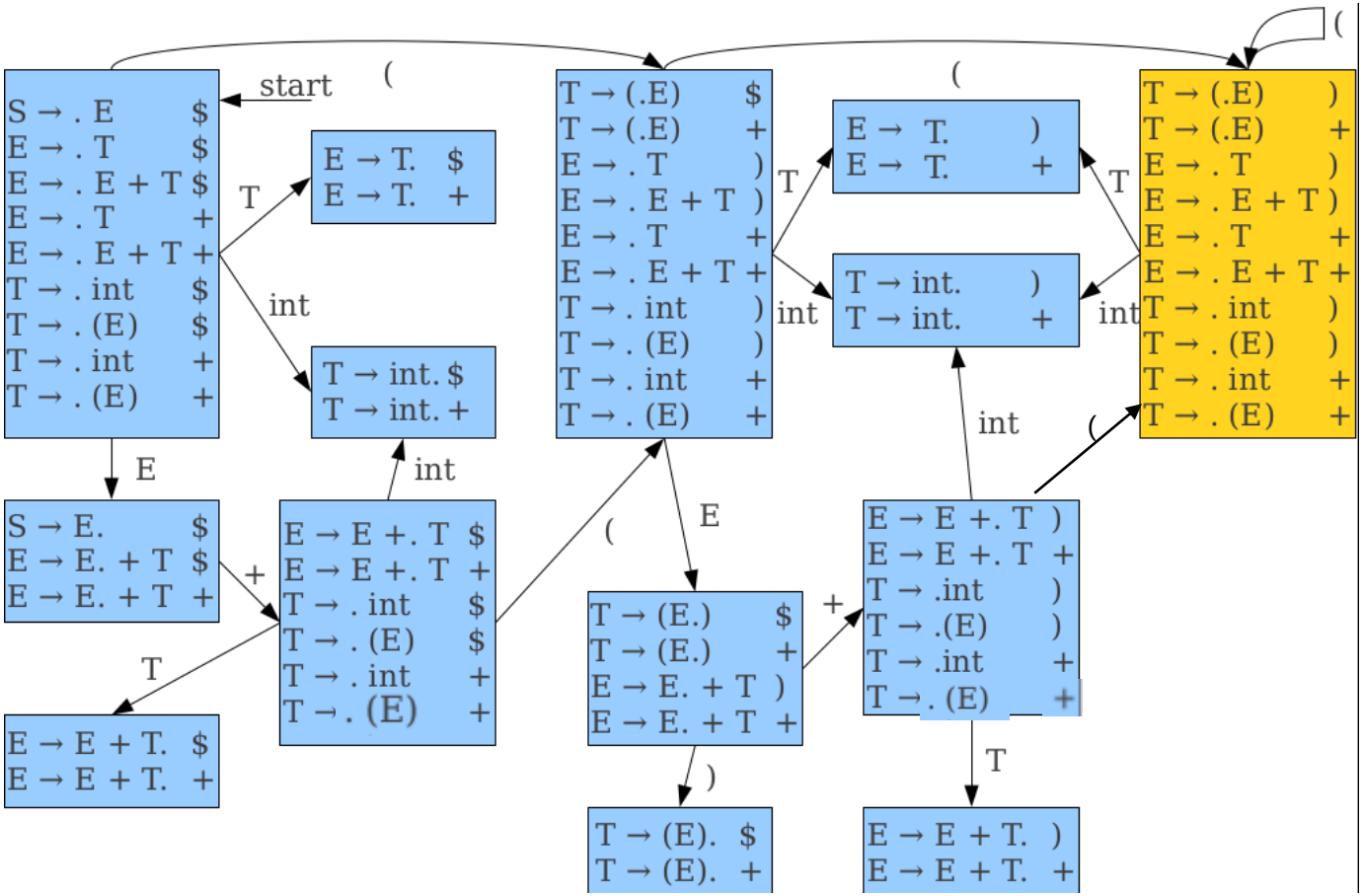
Deterministic LR(1) Automata



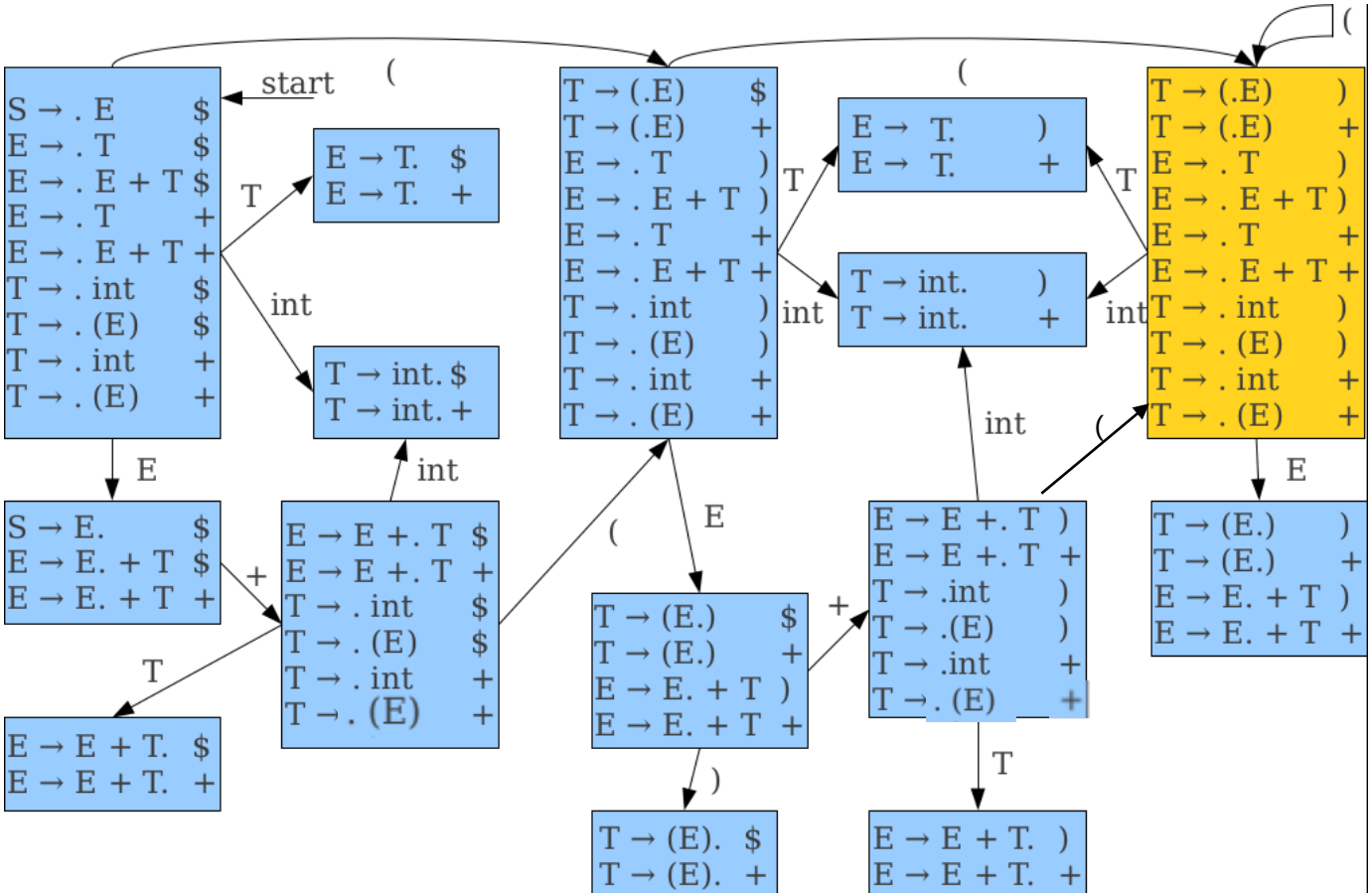
Deterministic LR(1) Automata



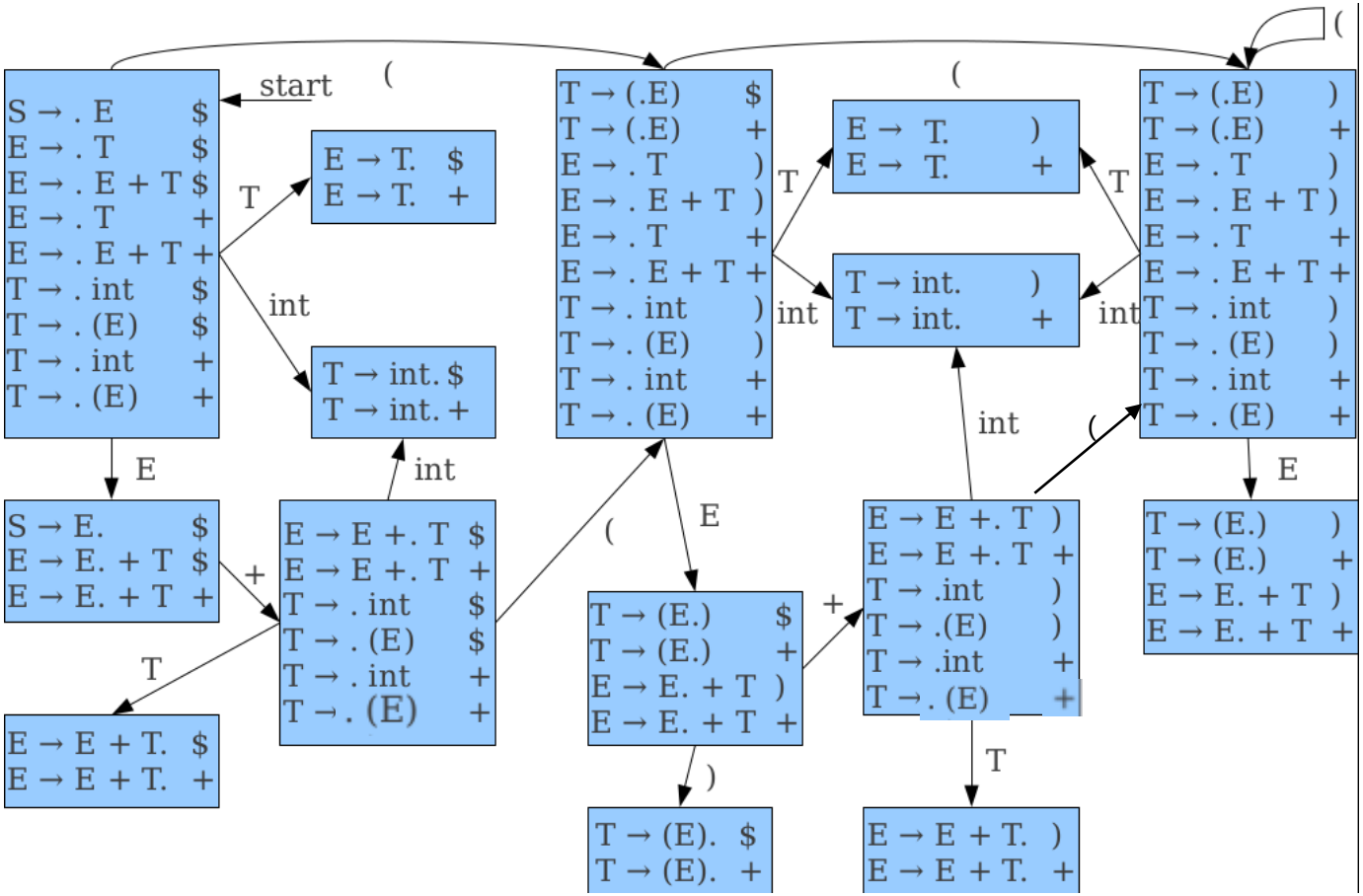
Deterministic LR(1) Automata



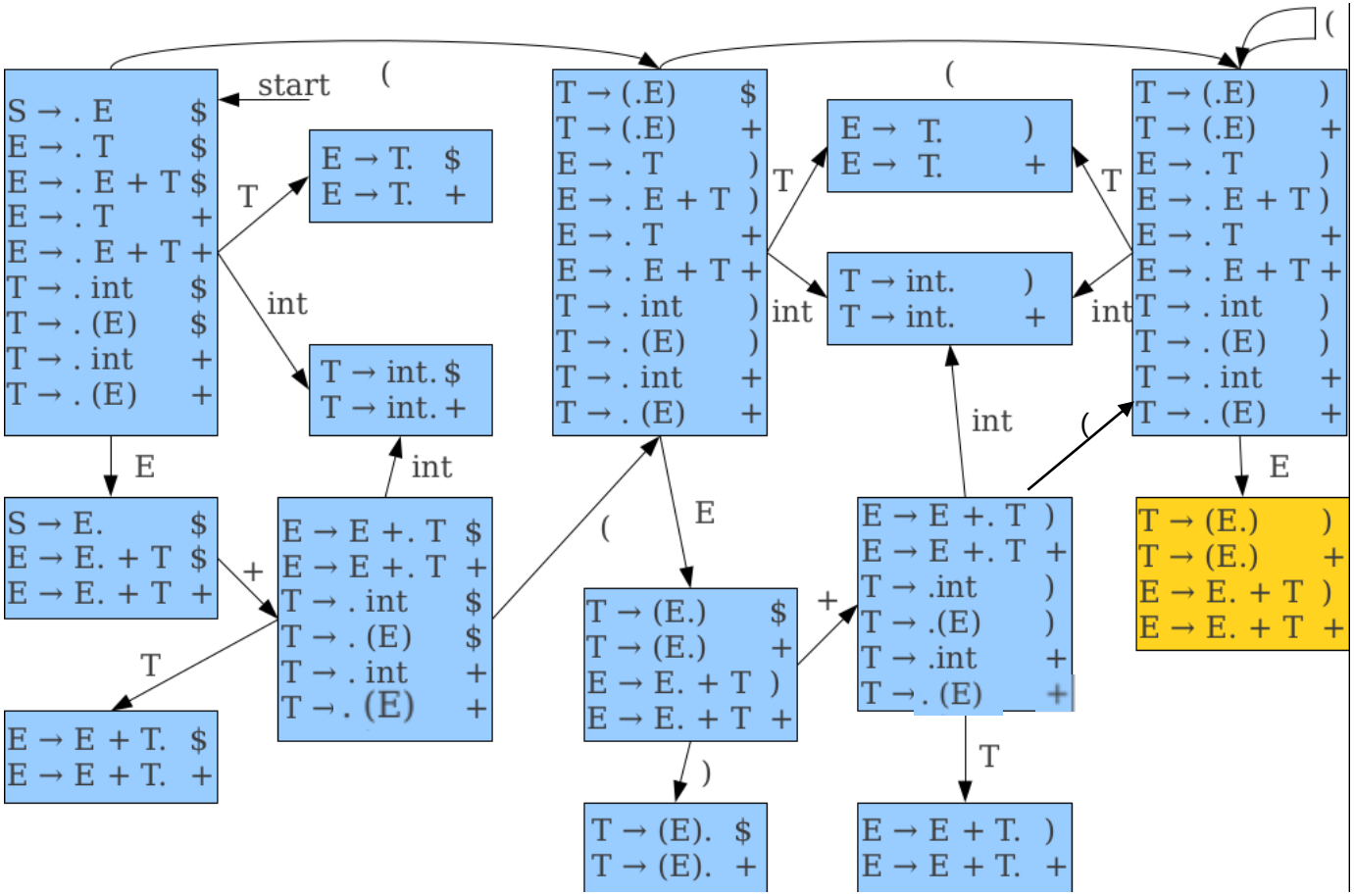
Deterministic LR(1) Automata



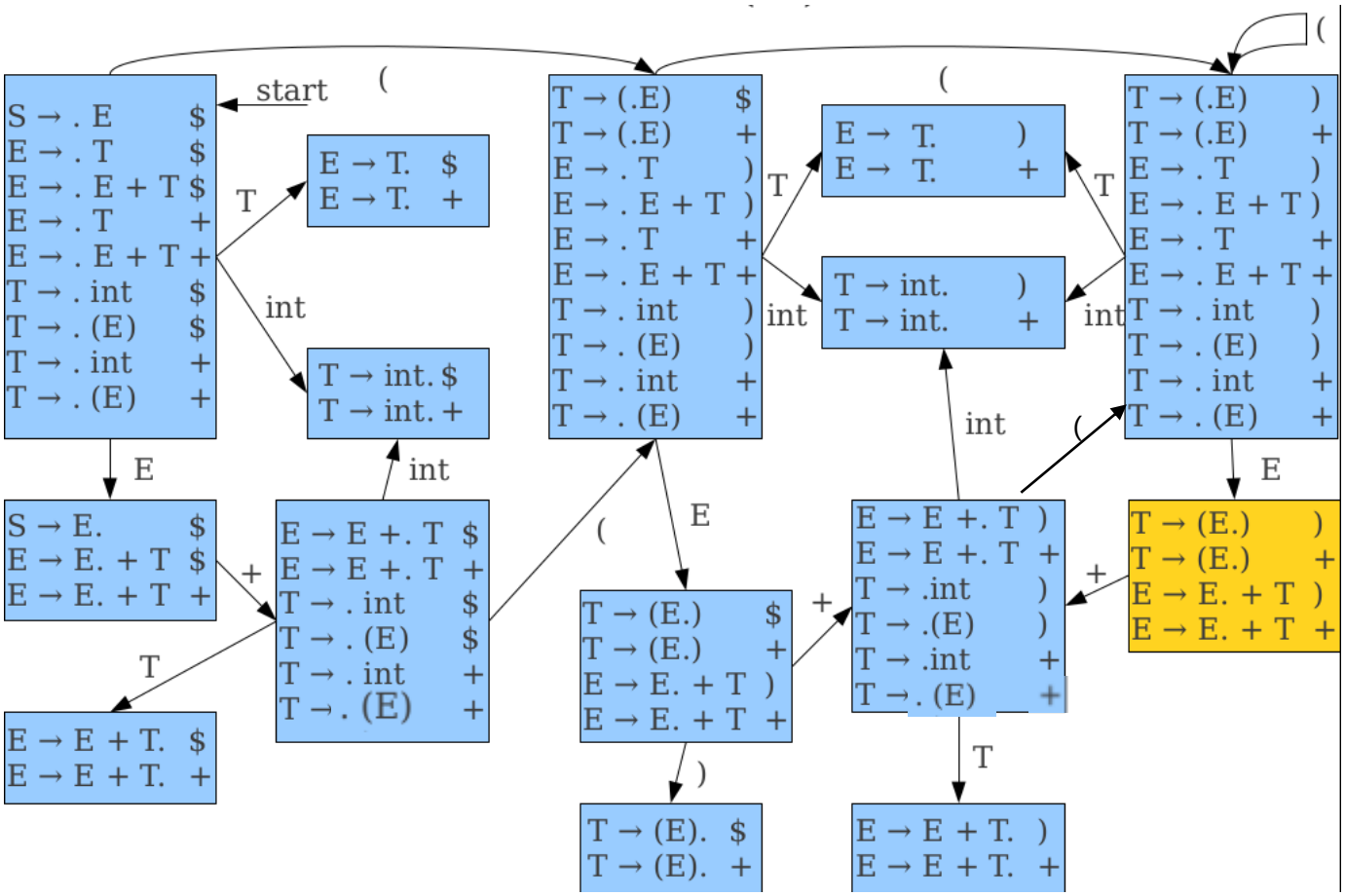
Deterministic LR(1) Automata



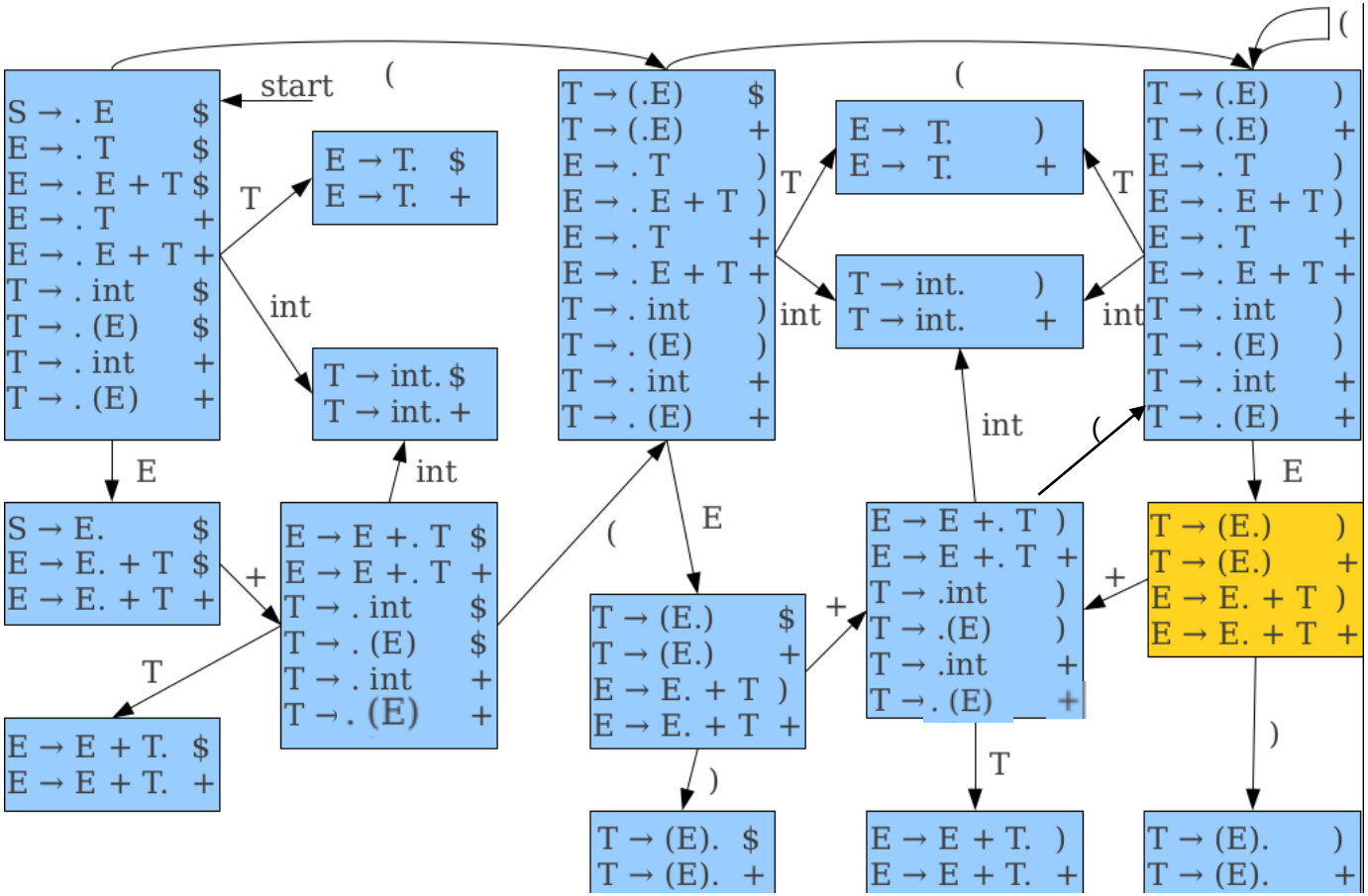
Deterministic LR(1) Automata



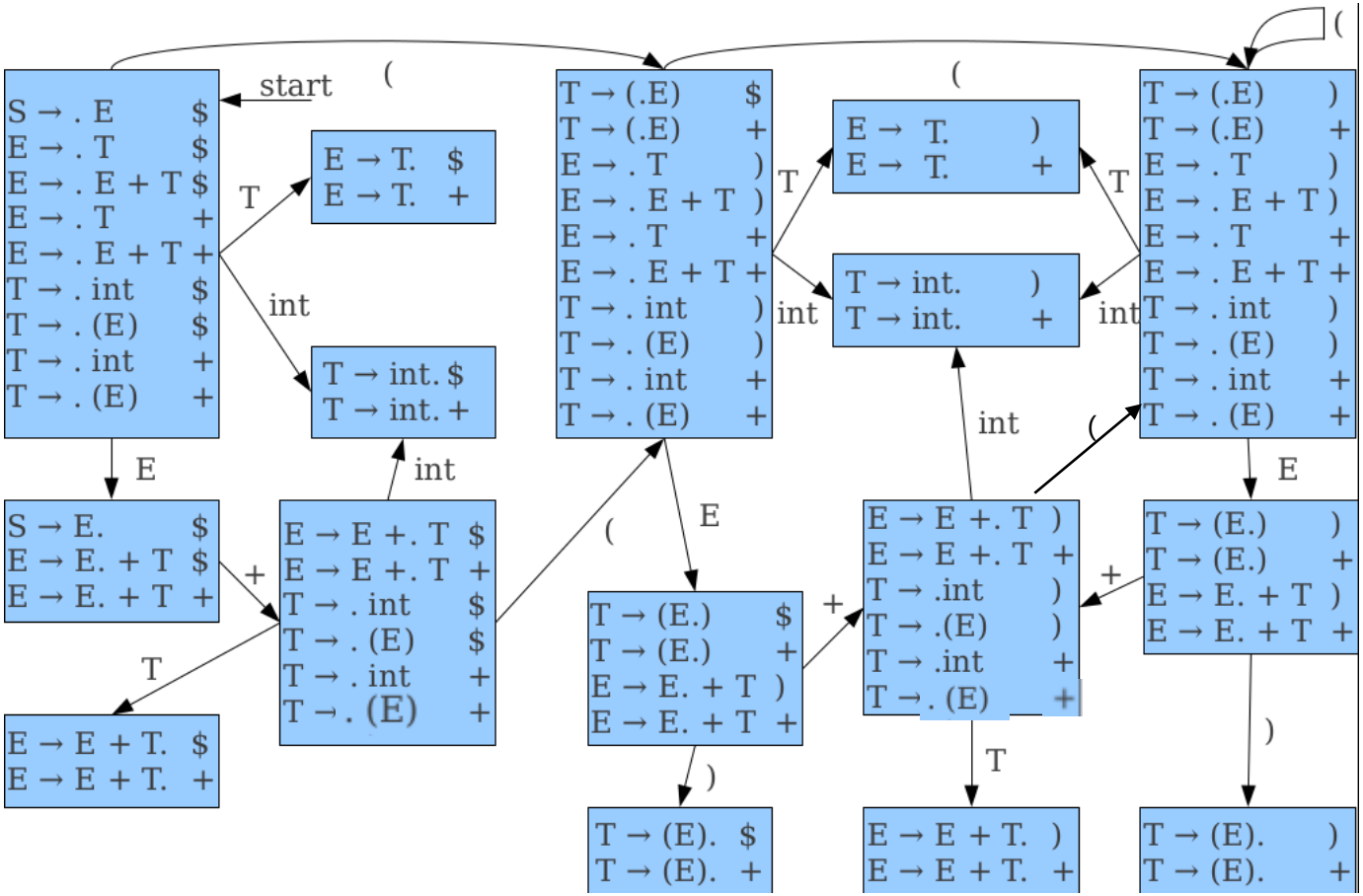
Deterministic LR(1) Automata



Deterministic LR(1) Automata



Deterministic LR(1) Automata



Constructing LR(1) Automata II

- Begin in a state containing $S \rightarrow \cdot\ E\ [\ \$\ ]$, where S is the start symbol.
- Compute the closure of the state:
  - If $A \rightarrow \alpha \cdot B\omega\ [t]$ is in the state, add $B \rightarrow \cdot\ \gamma\ [t]$ to the state for each production $B \rightarrow \gamma$ and for each terminal $t \in FIRST^*(\omega t)$
- Repeat until no new states are added:
  - If a state contains a production $A \rightarrow \alpha \cdot x\omega\ [t]$, add a transition on x from that state to the state containing the closure of $A \rightarrow \alpha x \cdot \omega\ [t]$.

Structure of LR(1) Automata

- Every LR(1) automaton simulates two processes simultaneously:
  - An LR(0) automaton for finding handles.
  - A lookahead tracker for determining what the lookahead is.
- Removing the lookaheads from an LR(1) automaton results in a LR(0) automaton for the same grammar.

Representing LR(1) Automata

- As with LR(0), use action and goto tables.
- goto table defined as before; encodes transition table as map from (state, token) to states.
- action table maps pairs (state, lookahead) to actions.
- Commonly combined into a single action/goto table.

LR(1) Table

S → E (1)
E → T (2)
E → E + T (3)
T → int (4)
T → (E) (5)

	int	(	)	+	$	T	E
1	s5					s4	s2
2				s6	ACCEPT		
3				r3	r3		
4				r2	r2		
5				r4	r4		
6	s5	s7				s3	
7	s11	s14				s10	s8
8			s9	s12			
9				r5	r5		
10			r2	r2			
11			r4	r4			
12	s11					s13	
13			r3	r3			
14	s11		s14			s10	s15
15			s16	s12			
16			r5	r5			

The LR(1) Parsing Algorithm

- Begin with an empty stack and the input set to ω$, where ω is the string to parse. Set state to the initial state.
- Repeat the following:
  - Let the next symbol of input be t.
  - If action[state, t] is shift, then shift the input and set state = goto[state, t].
  - If action[state, t] is reduce A → ω:
    - Pop |ω| symbols off the stack; replace them with A.
    - Let the state atop the stack be top-state.
    - Set state = goto[top-state, A]
  - If action[state, t] is accept, then the parse is done.
  - If action[state, t] is error, report an error.

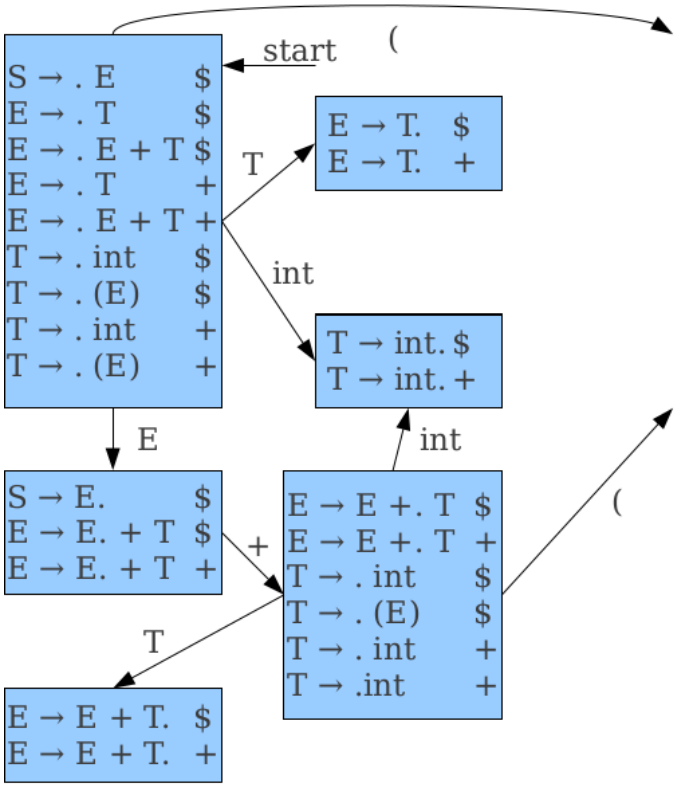
Constructing LR(1) Parse Tables

- For each state X:
  - If there is a production $A \rightarrow \omega \cdot [t]$, set

    action[X, t] = reduce $A \rightarrow \omega$.
  - If there is the special production $S \rightarrow E \cdot [\$]$, where S is the start symbol, set action[X, t] = accept.
  - If there is a transition out of s on symbol t, set

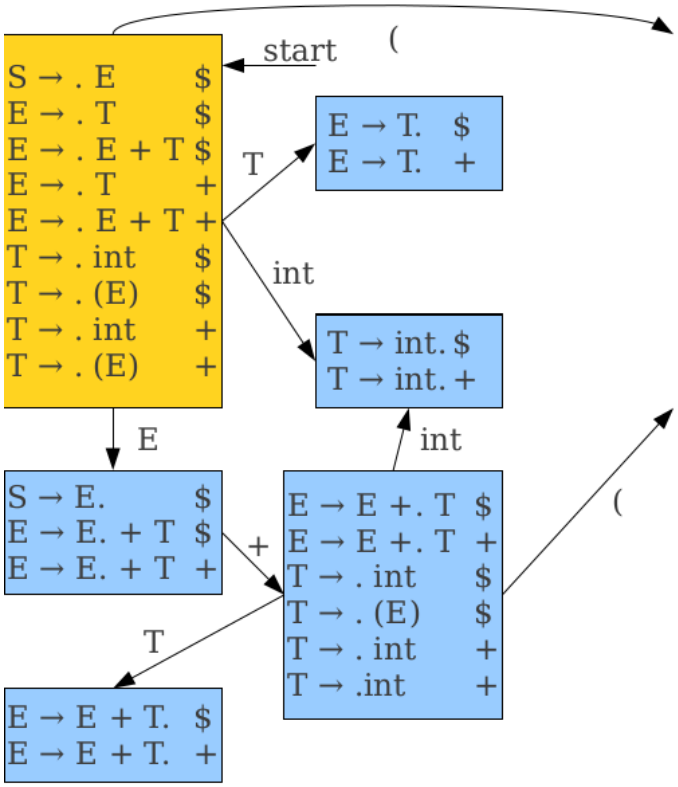
    action[X, t] = shift.
- Set all other actions to error.
- If any table entry contains two or more actions, the grammar is not LR(1).

Deterministic LR(1) Automata



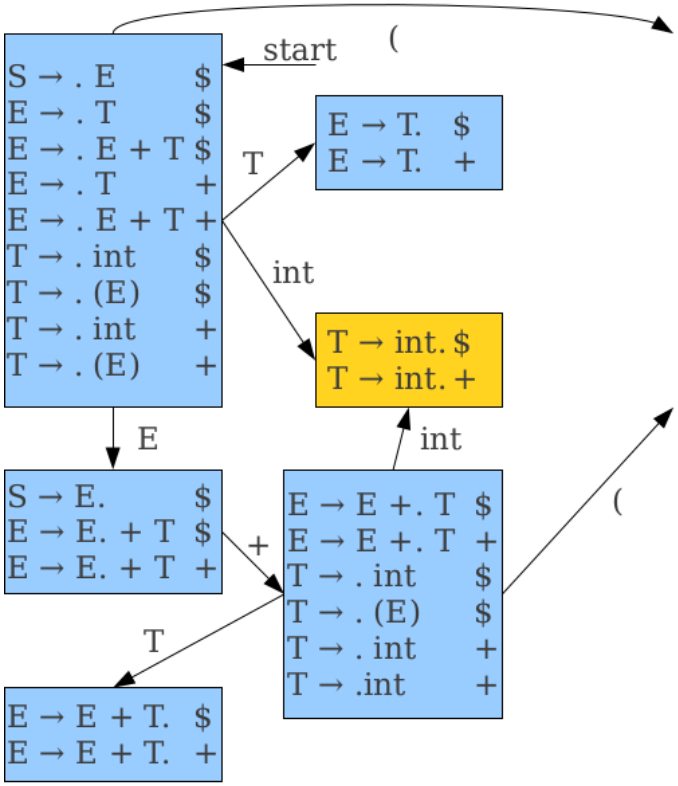
int	+	int	$

Deterministic LR(1) Automata



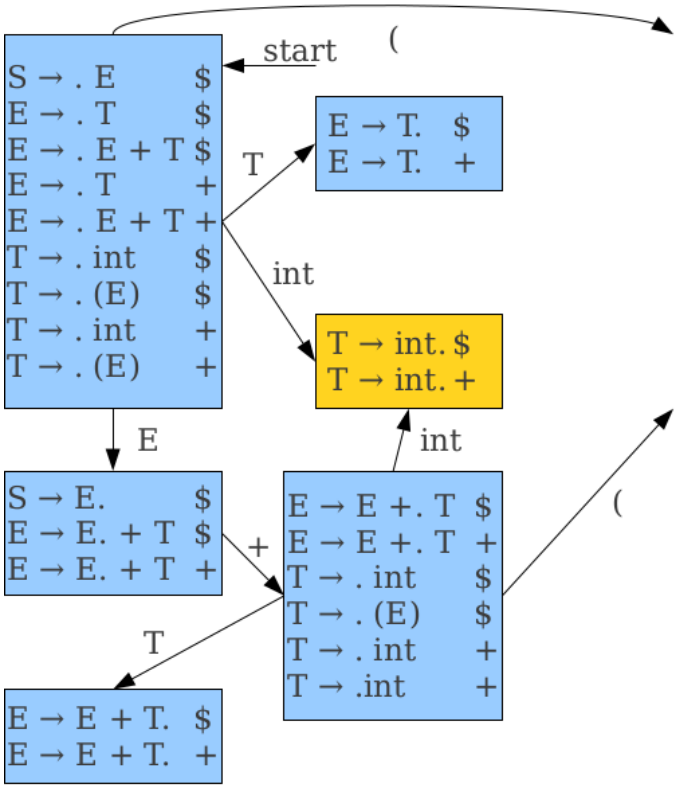
int	+	int	$

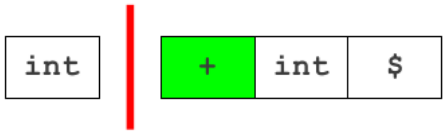
Deterministic LR(1) Automata



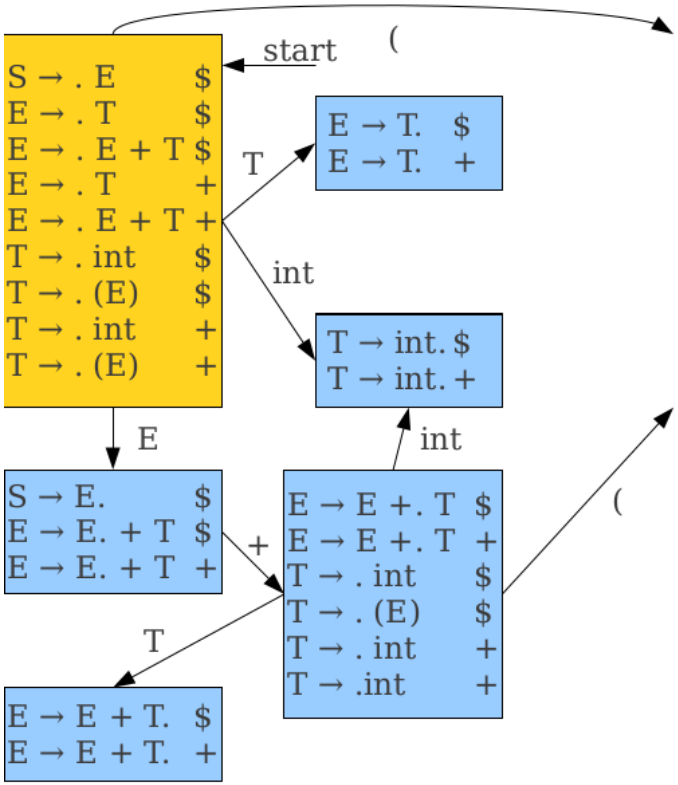
int	+	int	$

Deterministic LR(1) Automata



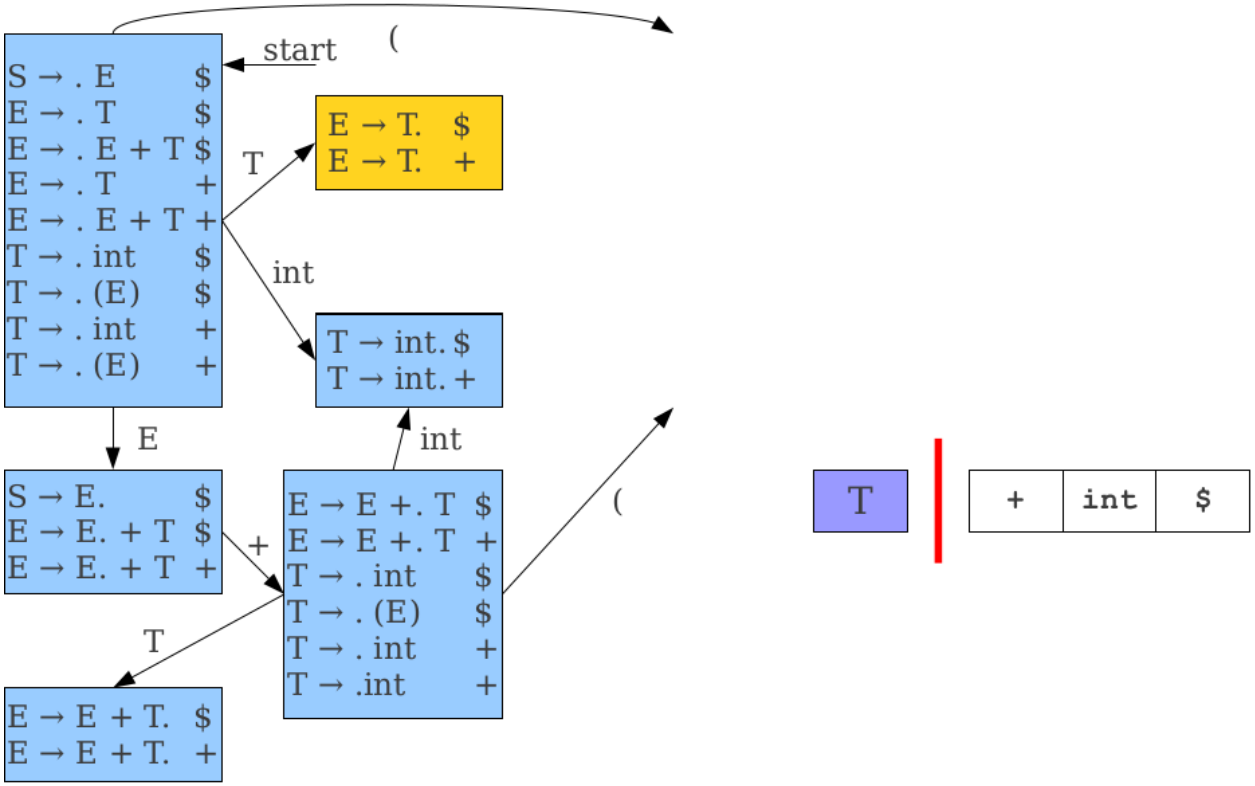


Deterministic LR(1) Automata

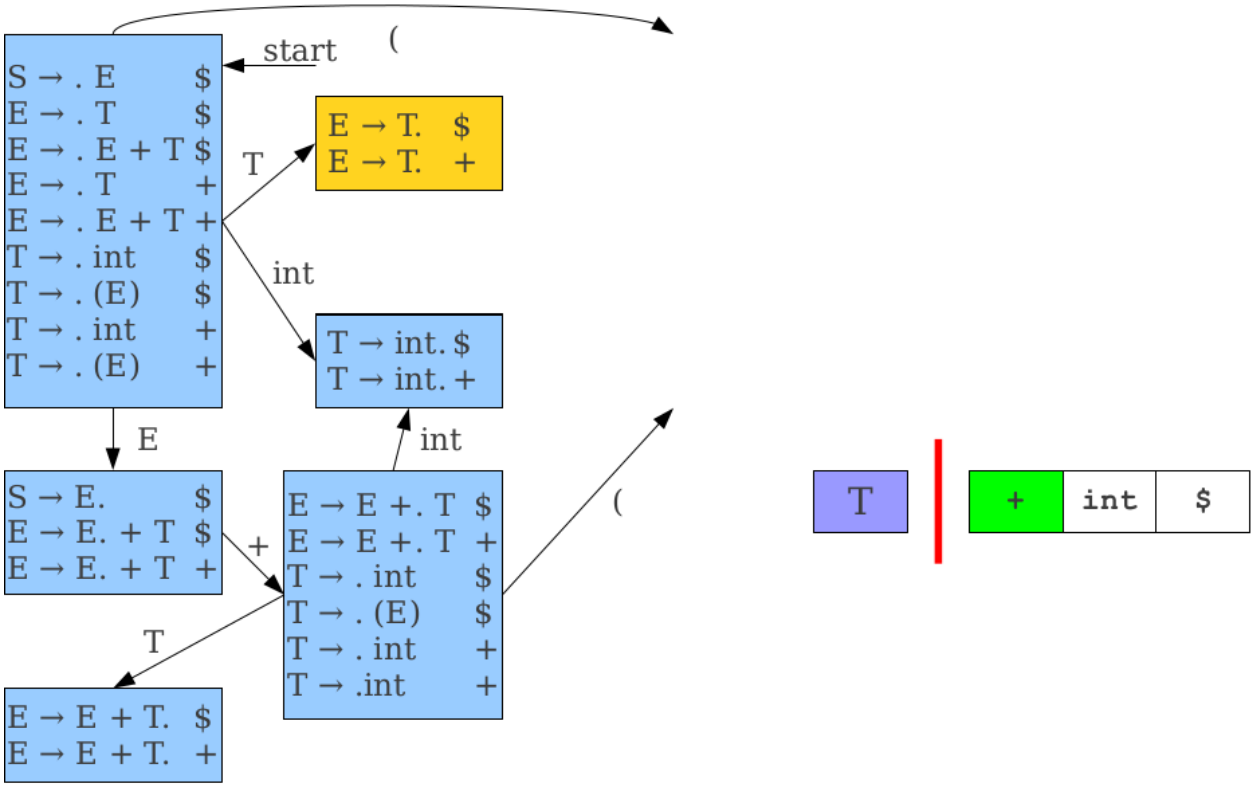


T	+	int	$

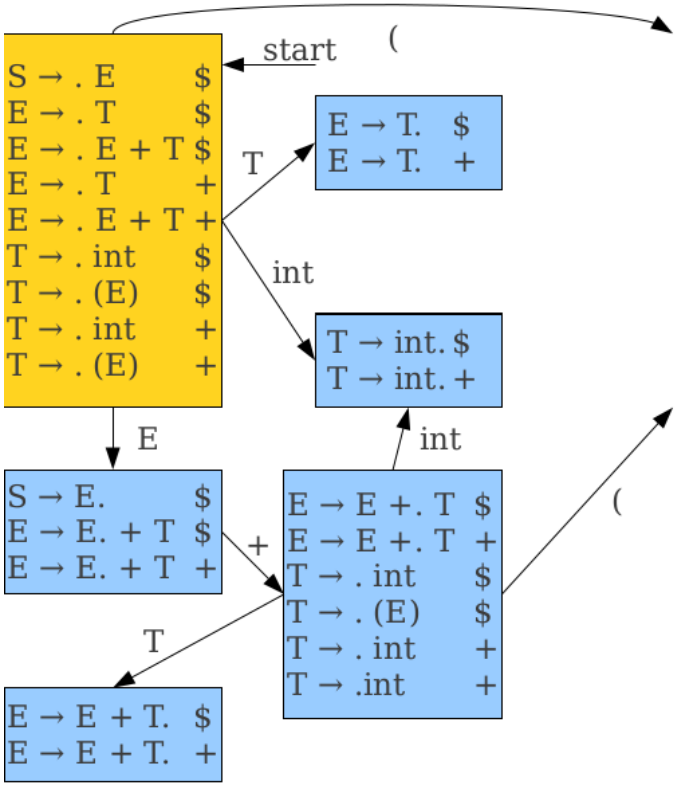
Deterministic LR(1) Automata

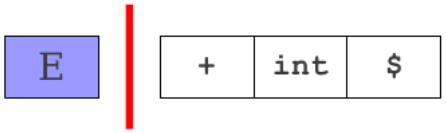


Deterministic LR(1) Automata

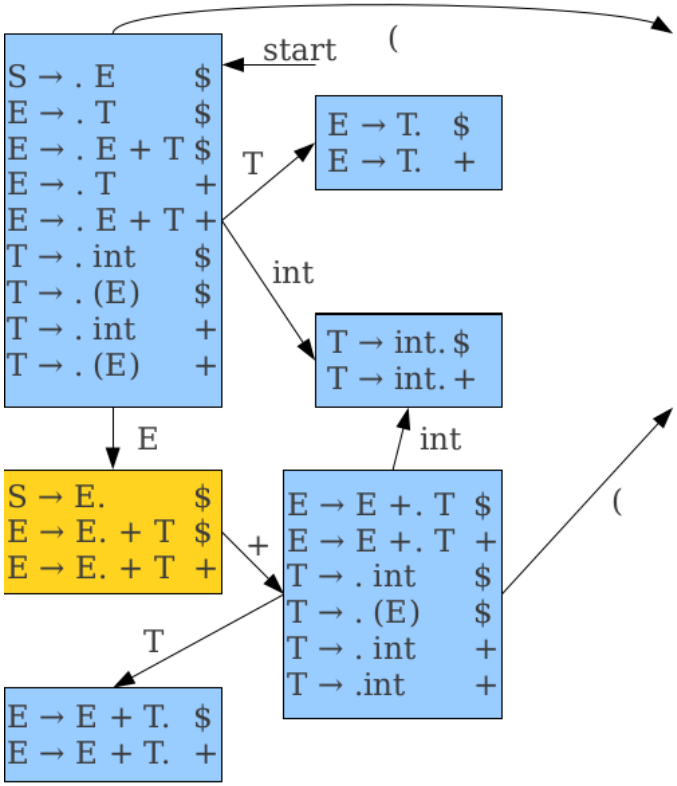


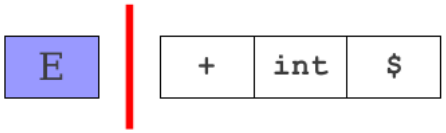
Deterministic LR(1) Automata



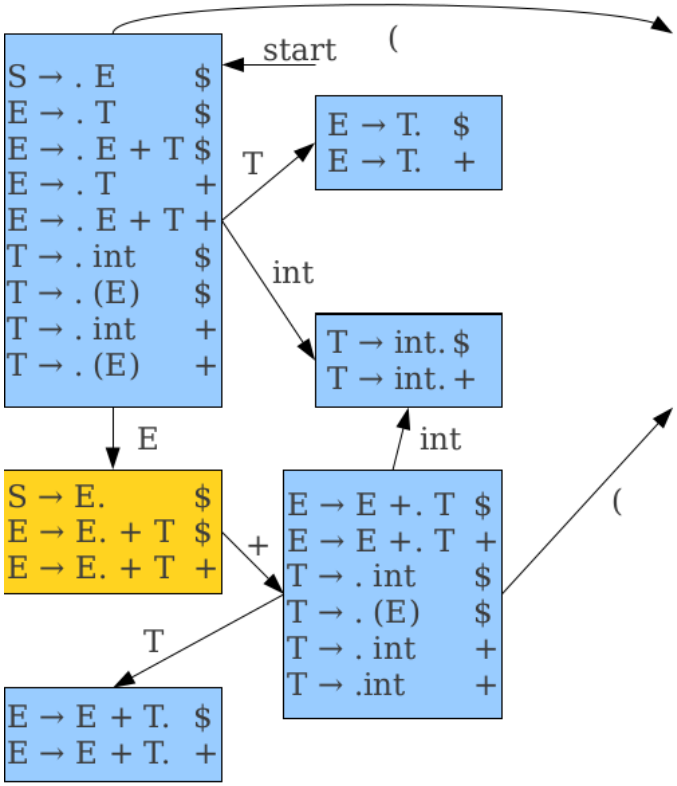


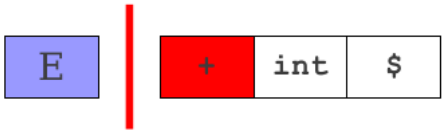
Deterministic LR(1) Automata



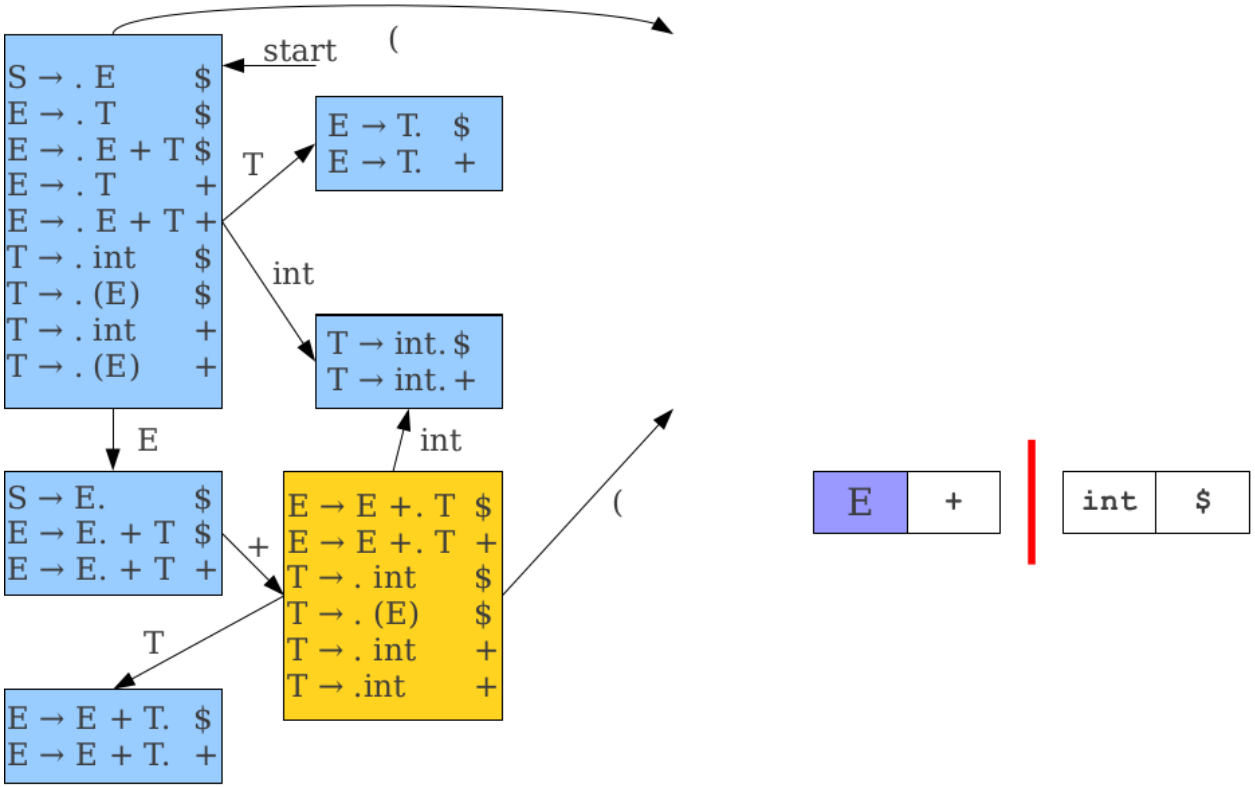


Deterministic LR(1) Automata

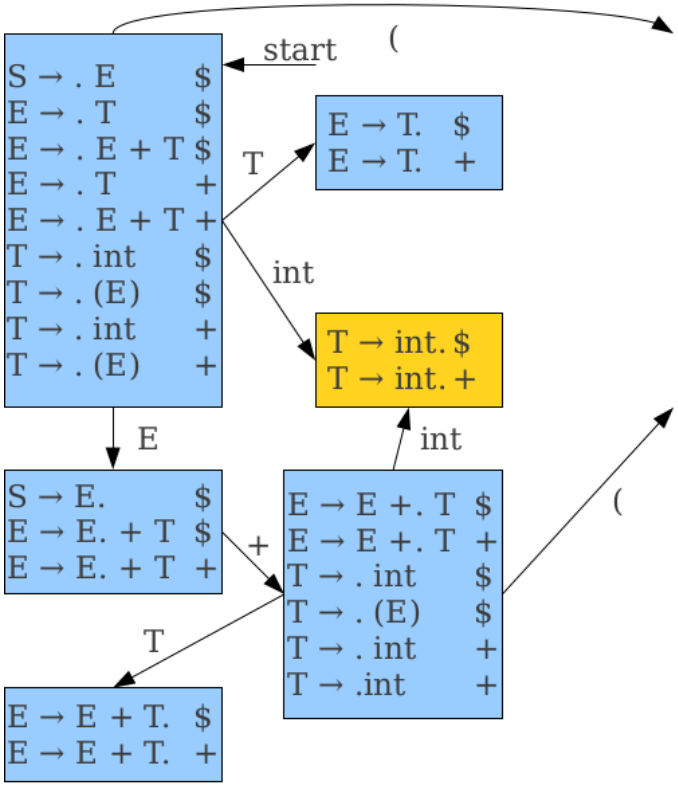


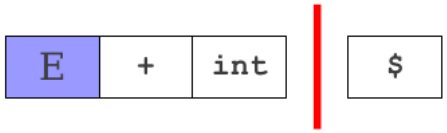


Deterministic LR(1) Automata

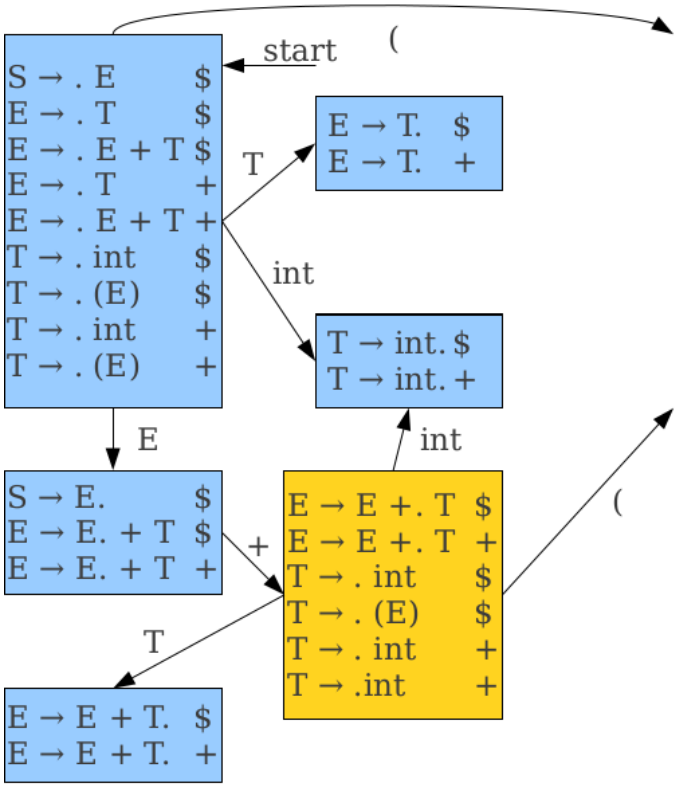


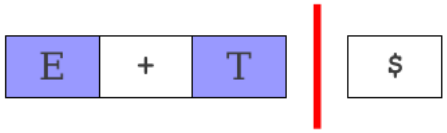
Deterministic LR(1) Automata



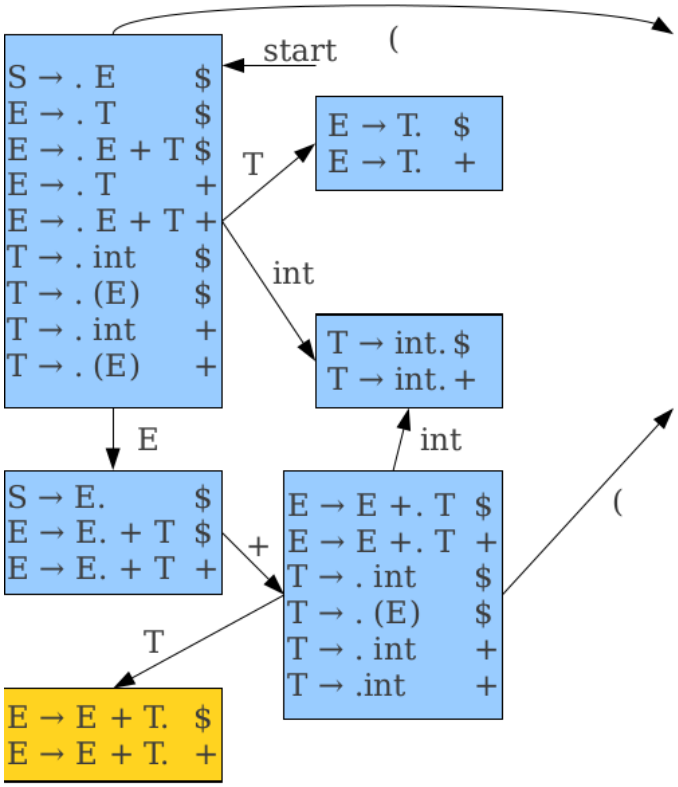


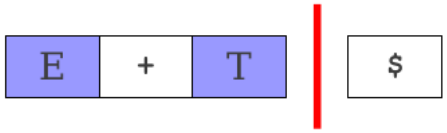
Deterministic LR(1) Automata



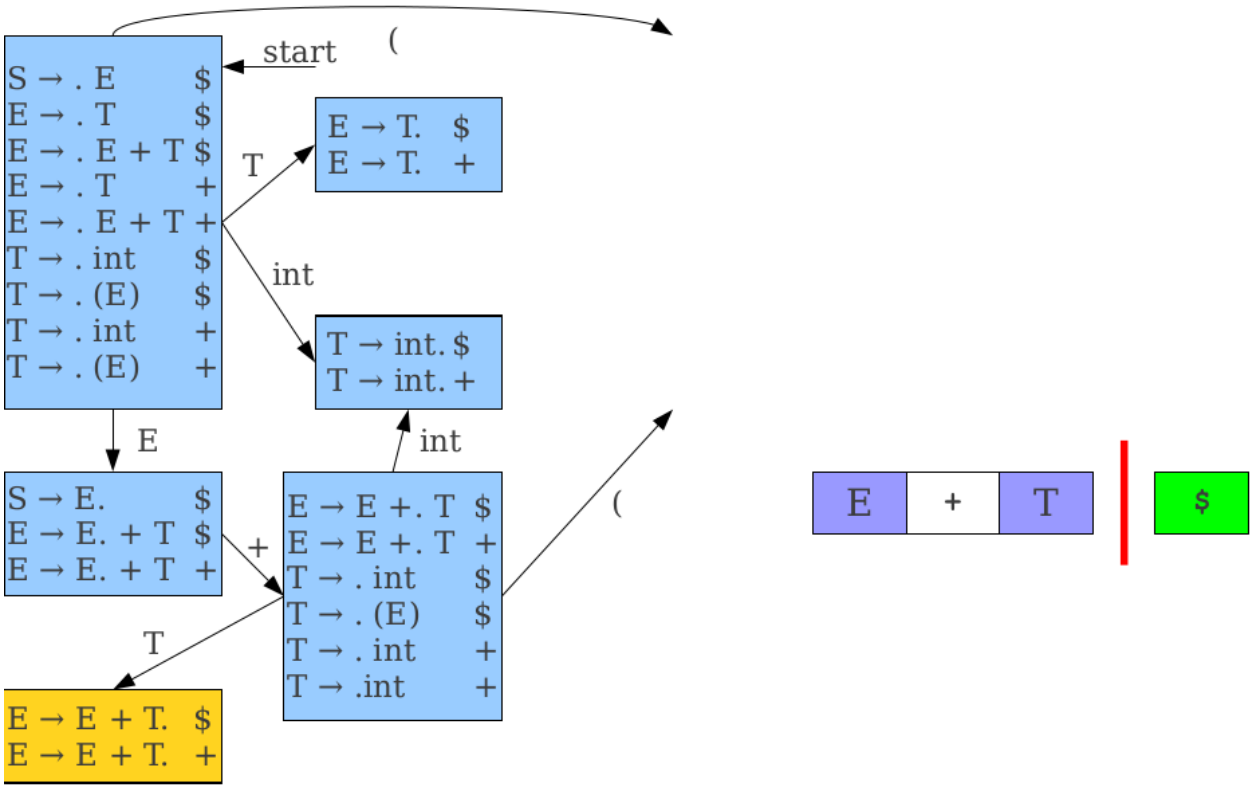


Deterministic LR(1) Automata

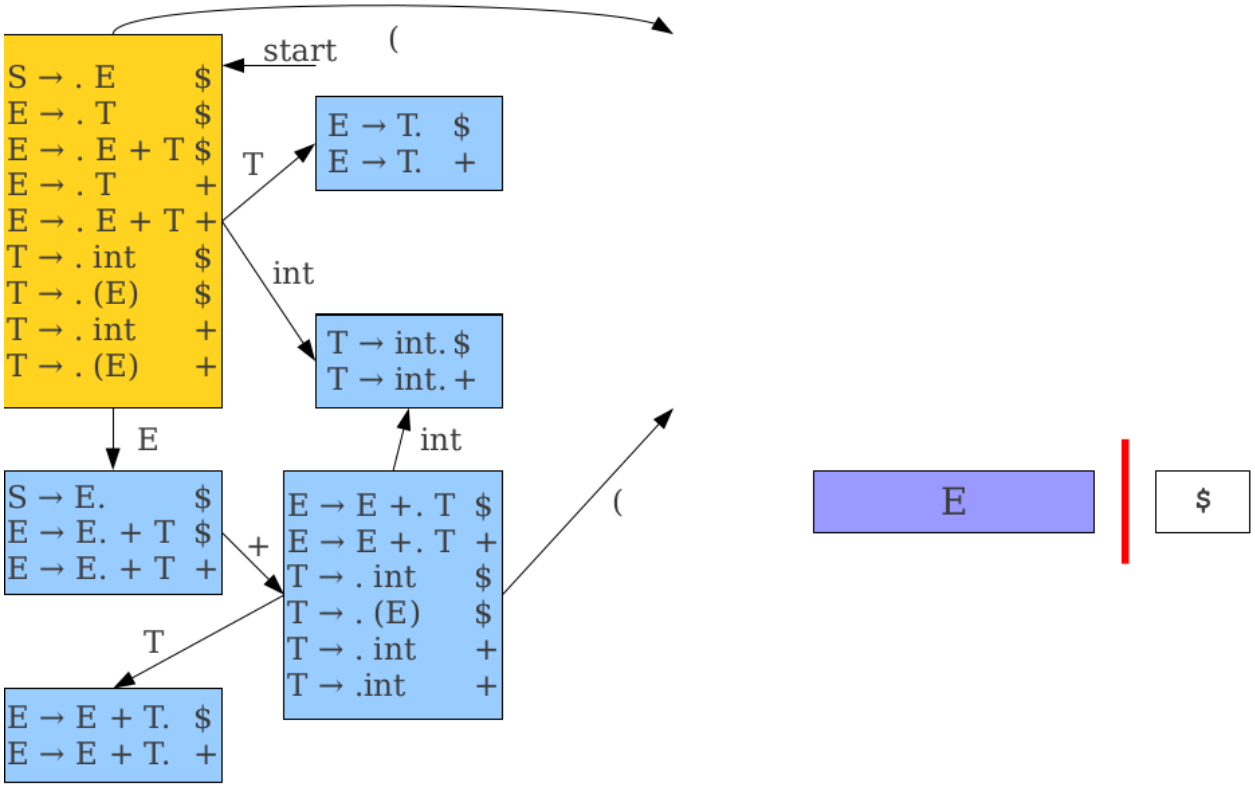




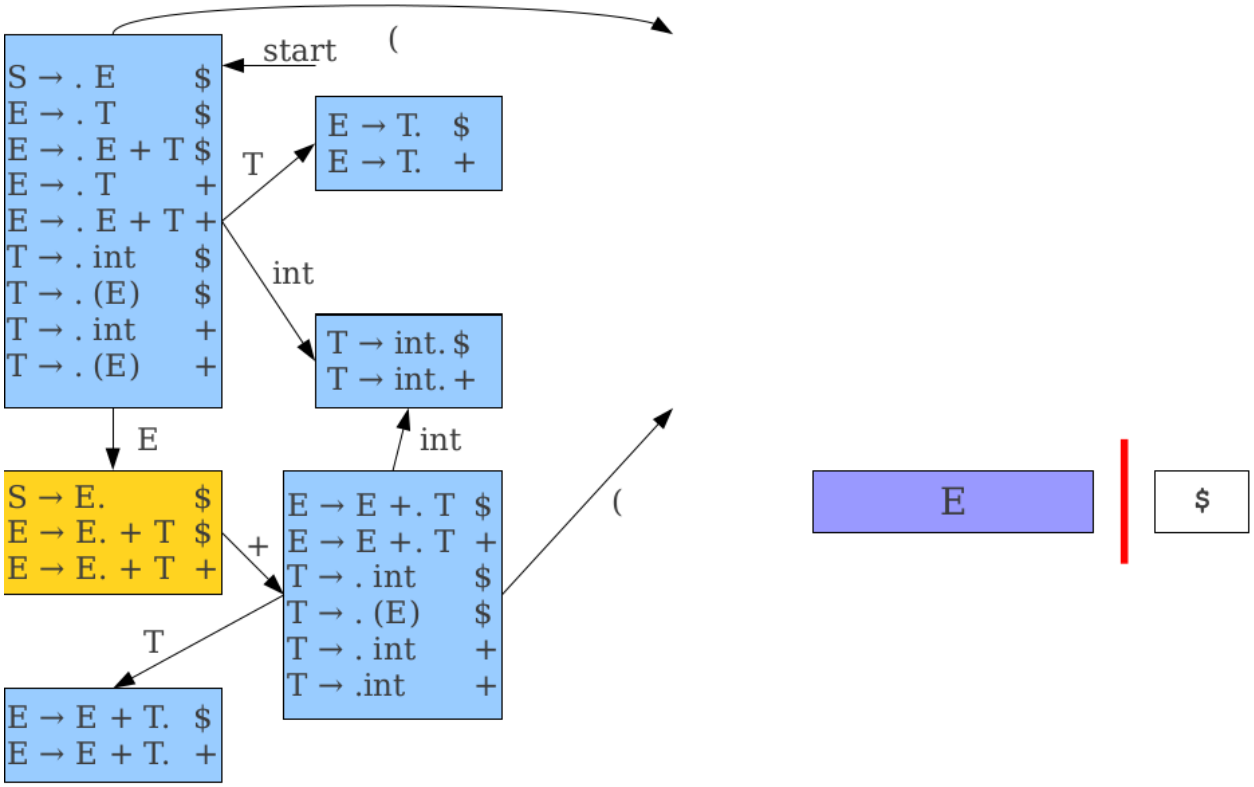
Deterministic LR(1) Automata



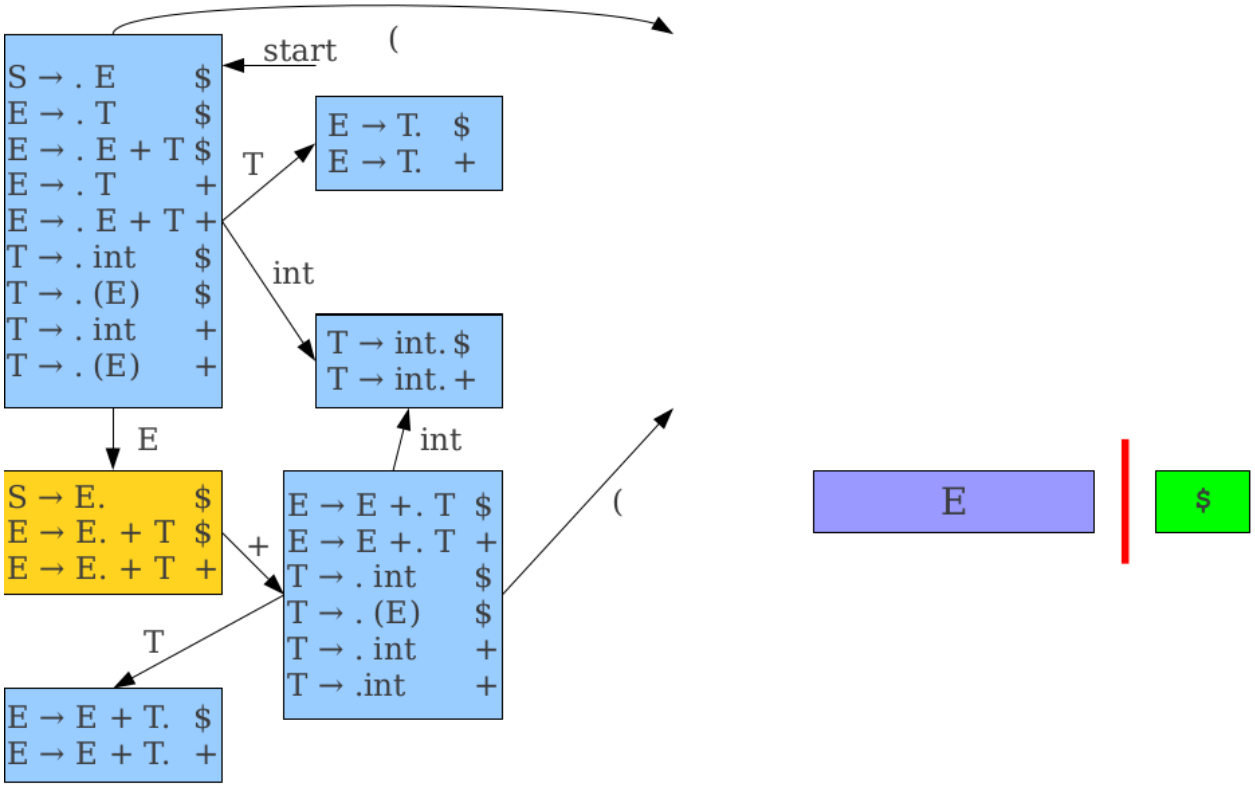
Deterministic LR(1) Automata



Deterministic LR(1) Automata



Deterministic LR(1) Automata



LR(1) Parser

Stack	Input	Action
1	int+(int+int)$	

	int	(	)	+	$	T	E
1	s5					s4	s2
2				s6	ACCEPT		
3				r3	r3		
4				r2	r2		
5				r4	r4		
6	s5	s7				s3	
7	s11	s14				s10	s8
8			s9	s12			
9				r5	r5		
10			r2	r2			
11			r4	r4			
12	s11					s13	
13			r3	r3			
14	s11		s14			s10	s15
15			s16	s12			
16			r5	r5			

- Input: int+(int+int)

$S \rightarrow E$ (1)
$E \rightarrow T$ (2)
$E \rightarrow E + T$ (3)
$T \rightarrow \texttt{int}$ (4)
$T \rightarrow (E)$ (5)

LR(1) Parser

Stack	Input	Action
1 1int5	int+(int+int)$ +(int+int)$	s5

	int	(	)	+	$	T	E
1	s5					s4	s2
2				s6	ACCEPT		
3				r3	r3		
4				r2	r2		
5				r4	r4		
6	s5	s7				s3	
7	s11	s14				s10	s8
8			s9	s12			
9				r5	r5		
10			r2	r2			
11			r4	r4			
12	s11					s13	
13			r3	r3			
14	s11		s14			s10	s15
15			s16	s12			
16			r5	r5			

- Input: int+(int+int)

$S \rightarrow E$ (1)
$E \rightarrow T$ (2)
$E \rightarrow E + T$ (3)
$T \rightarrow \texttt{int}$ (4)
$T \rightarrow (E)$ (5)

LR(1) Parser

Stack	Input	Action
1	int+(int+int)$	s5
1int5	+(int+int)$	r4
1T	+(int+int)$	

	int	(	)	+	$	T	E
1	s5					s4	s2
2				s6	ACCEPT		
3				r3	r3		
4				r2	r2		
5				r4	r4		
6	s5	s7				s3	
7	s11	s14				s10	s8
8			s9	s12			
9				r5	r5		
10			r2	r2			
11			r4	r4			
12	s11					s13	
13			r3	r3			
14	s11		s14			s10	s15
15			s16	s12			
16			r5	r5			

- Input: int+(int+int)

$S \rightarrow E$ (1)
$E \rightarrow T$ (2)
$E \rightarrow E + T$ (3)
$T \rightarrow \texttt{int}$ (4)
$T \rightarrow (E)$ (5)

LR(1) Parser

Stack	Input	Action
1	int+(int+int)$	s5
1int5	+(int+int)$	r4 (g4)
1T4	+(int+int)$	

	int	(	)	+	$	T	E
1	s5					s4	s2
2				s6	ACCEPT		
3				r3	r3		
4				r2	r2		
5				r4	r4		
6	s5	s7				s3	
7	s11	s14				s10	s8
8			s9	s12			
9				r5	r5		
10			r2	r2			
11			r4	r4			
12	s11					s13	
13			r3	r3			
14	s11		s14			s10	s15
15			s16	s12			
16			r5	r5			

- Input: int+(int+int)

$S \rightarrow E$ (1)
$E \rightarrow T$ (2)
$E \rightarrow E + T$ (3)
$T \rightarrow \texttt{int}$ (4)
$T \rightarrow (E)$ (5)

LR(1) Parser

Stack	Input	Action
1	int+(int+int)$	s5
1int5	+(int+int)$	r4 (g4)
1T4	+(int+int)$	r2
1E	+(int+int)$	

	int	(	)	+	$	T	E
1	s5					s4	s2
2				s6	ACCEPT		
3				r3	r3		
4				r2	r2		
5				r4	r4		
6	s5	s7				s3	
7	s11	s14				s10	s8
8			s9	s12			
9				r5	r5		
10			r2	r2			
11			r4	r4			
12	s11					s13	
13			r3	r3			
14	s11		s14			s10	s15
15			s16	s12			
16			r5	r5			

- Input: int+(int+int)

$S \rightarrow E$ (1)
$E \rightarrow T$ (2)
$E \rightarrow E + T$ (3)
$T \rightarrow \texttt{int}$ (4)
$T \rightarrow (E)$ (5)

LR(1) Parser

Stack	Input	Action
1	int+(int+int)$	s5
1int5	+(int+int)$	r4 (g4)
1T4	+(int+int)$	r2 (g2)
1E2	+(int+int)$	

	int	(	)	+	$	T	E
1	s5					s4	s2
2				s6	ACCEPT		
3				r3	r3		
4				r2	r2		
5				r4	r4		
6	s5	s7				s3	
7	s11	s14				s10	s8
8			s9	s12			
9				r5	r5		
10			r2	r2			
11			r4	r4			
12	s11					s13	
13			r3	r3			
14	s11		s14			s10	s15
15			s16	s12			
16			r5	r5			

- Input: int+(int+int)

$S \rightarrow E$ (1)
$E \rightarrow T$ (2)
$E \rightarrow E + T$ (3)
$T \rightarrow \texttt{int}$ (4)
$T \rightarrow (E)$ (5)

LR(1) Parser

Stack	Input	Action
1	int+(int+int)$	s5
1int5	+(int+int)$	r4 (g4)
1T4	+(int+int)$	r2 (g2)
1E2	+(int+int)$	s6
1E2+6	(int+int)$	

	int	(	)	+	$	T	E
1	s5					s4	s2
2				s6	ACCEPT		
3				r3	r3		
4				r2	r2		
5				r4	r4		
6	s5	s7				s3	
7	s11	s14				s10	s8
8			s9	s12			
9				r5	r5		
10			r2	r2			
11			r4	r4			
12	s11					s13	
13			r3	r3			
14	s11		s14			s10	s15
15			s16	s12			
16			r5	r5			

- Input: int+(int+int)

$S \rightarrow E$ (1)
$E \rightarrow T$ (2)
$E \rightarrow E + T$ (3)
$T \rightarrow \texttt{int}$ (4)
$T \rightarrow (E)$ (5)

LR(1) Parser

Stack	Input	Action
1	int+(int+int)$	s5
1int5	+(int+int)$	r4 (g4)
1T4	+(int+int)$	r2 (g2)
1E2	+(int+int)$	s6
1E2+6	(int+int)$	s7
1E2+6(7	int+int)$	

	int	(	)	+	$	T	E
1	s5					s4	s2
2				s6	ACCEPT		
3				r3	r3		
4				r2	r2		
5				r4	r4		
6	s5	s7				s3	
7	s11	s14				s10	s8
8			s9	s12			
9				r5	r5		
10			r2	r2			
11			r4	r4			
12	s11					s13	
13			r3	r3			
14	s11		s14			s10	s15
15			s16	s12			
16			r5	r5			

- Input: int+(int+int)

$S \rightarrow E$ (1)
$E \rightarrow T$ (2)
$E \rightarrow E + T$ (3)
$T \rightarrow \texttt{int}$ (4)
$T \rightarrow (E)$ (5)

LR(1) Parser

Stack	Input	Action
1	int+(int+int)$	s5
1int5	+(int+int)$	r4 (g4)
1T4	+(int+int)$	r2 (g2)
1E2	+(int+int)$	s6
1E2+6	(int+int)$	s7
1E2+6(7	int+int)$	s11
1E2+6(7int11	+int)$	

	int	(	)	+	$	T	E
1	s5					s4	s2
2				s6	ACCEPT		
3				r3	r3		
4				r2	r2		
5				r4	r4		
6	s5	s7				s3	
7	s11	s14				s10	s8
8			s9	s12			
9				r5	r5		
10			r2	r2			
11			r4	r4			
12	s11					s13	
13			r3	r3			
14	s11		s14			s10	s15
15			s16	s12			
16			r5	r5			

- Input: int+(int+int)

$S \rightarrow E$ (1)
$E \rightarrow T$ (2)
$E \rightarrow E + T$ (3)
$T \rightarrow \texttt{int}$ (4)
$T \rightarrow (E)$ (5)

LR(1) Parser

Stack	Input	Action
1	int+(int+int)$	s5
1int5	+(int+int)$	r4 (g4)
1T4	+(int+int)$	r2 (g2)
1E2	+(int+int)$	s6
1E2+6	(int+int)$	s7
1E2+6(7	int+int)$	s11
1E2+6(7int11	+int)$	r4
1E2+6(7T	+int)$	

	int	(	)	+	$	T	E
1	s5					s4	s2
2				s6	ACCEPT		
3				r3	r3		
4				r2	r2		
5				r4	r4		
6	s5	s7				s3	
7	s11	s14				s10	s8
8			s9	s12			
9				r5	r5		
10			r2	r2			
11			r4	r4			
12	s11					s13	
13			r3	r3			
14	s11		s14			s10	s15
15			s16	s12			
16			r5	r5			

- Input: int+(int+int)

$S \rightarrow E$ (1)
$E \rightarrow T$ (2)
$E \rightarrow E + T$ (3)
$T \rightarrow \texttt{int}$ (4)
$T \rightarrow (E)$ (5)

LR(1) Parser

Stack	Input	Action
1	int+(int+int)$	s5
1int5	+(int+int)$	r4 (g4)
1T4	+(int+int)$	r2 (g2)
1E2	+(int+int)$	s6
1E2+6	(int+int)$	s7
1E2+6(7	int+int)$	s11
1E2+6(7int11	+int)$	r4 (g10)
1E2+6(7T10	+int)$	

	int	(	)	+	$	T	E
1	s5					s4	s2
2				s6	ACCEPT		
3				r3	r3		
4				r2	r2		
5				r4	r4		
6	s5	s7				s3	
7	s11	s14				s10	s8
8			s9	s12			
9				r5	r5		
10			r2	r2			
11			r4	r4			
12	s11					s13	
13			r3	r3			
14	s11		s14			s10	s15
15			s16	s12			
16			r5	r5			

- Input: int+(int+int)

$S \rightarrow E$ (1)
$E \rightarrow T$ (2)
$E \rightarrow E + T$ (3)
$T \rightarrow \texttt{int}$ (4)
$T \rightarrow (E)$ (5)

LR(1) Parser

Stack	Input	Action
1	int+(int+int)$	s5
1int5	+(int+int)$	r4 (g4)
1T4	+(int+int)$	r2 (g2)
1E2	+(int+int)$	s6
1E2+6	(int+int)$	s7
1E2+6(7	int+int)$	s11
1E2+6(7int11	+int)$	r4 (g10)
1E2+6(7T10	+int)$	r2
1E2+6(7E	+int)$	

	int	(	)	+	$	T	E
1	s5					s4	s2
2				s6	ACCEPT		
3				r3	r3		
4				r2	r2		
5				r4	r4		
6	s5	s7				s3	
7	s11	s14				s10	s8
8			s9	s12			
9				r5	r5		
10			r2	r2			
11			r4	r4			
12	s11					s13	
13			r3	r3			
14	s11		s14			s10	s15
15			s16	s12			
16			r5	r5			

- Input: int+(int+int)

$S \rightarrow E$ (1)
$E \rightarrow T$ (2)
$E \rightarrow E + T$ (3)
$T \rightarrow \texttt{int}$ (4)
$T \rightarrow (E)$ (5)

LR(1) Parser

Stack	Input	Action
1	int+(int+int)$	s5
1int5	+(int+int)$	r4 (g4)
1T4	+(int+int)$	r2 (g2)
1E2	+(int+int)$	s6
1E2+6	(int+int)$	s7
1E2+6(7	int+int)$	s11
1E2+6(7int11	+int)$	r4 (g10)
1E2+6(7T10	+int)$	r2 (g8)
1E2+6(7E8	+int)$	

	int	(	)	+	$	T	E
1	s5					s4	s2
2				s6	ACCEPT		
3				r3	r3		
4				r2	r2		
5				r4	r4		
6	s5	s7				s3	
7	s11	s14				s10	s8
8			s9	s12			
9				r5	r5		
10			r2	r2			
11			r4	r4			
12	s11					s13	
13			r3	r3			
14	s11		s14			s10	s15
15			s16	s12			
16			r5	r5			

- Input: int+(int+int)

$S \rightarrow E$ (1)
$E \rightarrow T$ (2)
$E \rightarrow E + T$ (3)
$T \rightarrow \texttt{int}$ (4)
$T \rightarrow (E)$ (5)

LR(1) Parser

Stack	Input	Action
1	int+(int+int)$	s5
1int5	+(int+int)$	r4 (g4)
1T4	+(int+int)$	r2 (g2)
1E2	+(int+int)$	s6
1E2+6	(int+int)$	s7
1E2+6(7	int+int)$	s11
1E2+6(7int11	+int)$	r4 (g10)
1E2+6(7T10	+int)$	r2 (g8)
1E2+6(7E8	+int)$	s12
1E2+6(7E8+12	int)$	

	int	(	)	+	$	T	E
1	s5					s4	s2
2				s6	ACCEPT		
3				r3	r3		
4				r2	r2		
5				r4	r4		
6	s5	s7				s3	
7	s11	s14				s10	s8
8			s9	s12			
9				r5	r5		
10			r2	r2			
11			r4	r4			
12	s11					s13	
13			r3	r3			
14	s11		s14			s10	s15
15			s16	s12			
16			r5	r5			

- Input: int+(int+int)

$S \rightarrow E$ (1)
$E \rightarrow T$ (2)
$E \rightarrow E + T$ (3)
$T \rightarrow \texttt{int}$ (4)
$T \rightarrow (E)$ (5)

LR(1) Parser

Stack	Input	Action
1	int+(int+int)$	s5
1int5	+(int+int)$	r4 (g4)
1T4	+(int+int)$	r2 (g2)
1E2	+(int+int)$	s6
1E2+6	(int+int)$	s7
1E2+6(7	int+int)$	s11
1E2+6(7int11	+int)$	r4 (g10)
1E2+6(7T10	+int)$	r2 (g8)
1E2+6(7E8	+int)$	s12
1E2+6(7E8+12	int)$	s11
1E2+6(7E8+12int11	)$	

	int	(	)	+	$	T	E
1	s5					s4	s2
2				s6	ACCEPT		
3				r3	r3		
4				r2	r2		
5				r4	r4		
6	s5	s7				s3	
7	s11	s14				s10	s8
8			s9	s12			
9				r5	r5		
10			r2	r2			
11			r4	r4			
12	s11					s13	
13			r3	r3			
14	s11		s14			s10	s15
15			s16	s12			
16			r5	r5			

- Input: int+(int+int)

$S \rightarrow E$ (1)
$E \rightarrow T$ (2)
$E \rightarrow E + T$ (3)
$T \rightarrow \texttt{int}$ (4)
$T \rightarrow (E)$ (5)

LR(1) Parser

Stack	Input	Action
1	int+(int+int)$	s5
1int5	+(int+int)$	r4 (g4)
1T4	+(int+int)$	r2 (g2)
1E2	+(int+int)$	s6
1E2+6	(int+int)$	s7
1E2+6(7	int+int)$	s11
1E2+6(7int11	+int)$	r4 (g10)
1E2+6(7T10	+int)$	r2 (g8)
1E2+6(7E8	+int)$	s12
1E2+6(7E8+12	int)$	s11
1E2+6(7E8+12int11	)$	r4
1E2+6(7E8+12T	)$	

	int	(	)	+	$	T	E
1	s5					s4	s2
2				s6	ACCEPT		
3				r3	r3		
4				r2	r2		
5				r4	r4		
6	s5	s7				s3	
7	s11	s14				s10	s8
8			s9	s12			
9				r5	r5		
10			r2	r2			
11			r4	r4			
12	s11					s13	
13			r3	r3			
14	s11		s14			s10	s15
15			s16	s12			
16			r5	r5			

- Input: int+(int+int)

$S \rightarrow E$ (1)
$E \rightarrow T$ (2)
$E \rightarrow E + T$ (3)
$T \rightarrow \texttt{int}$ (4)
$T \rightarrow (E)$ (5)

LR(1) Parser

Stack	Input	Action
1	int+(int+int)$	s5
1int5	+(int+int)$	r4 (g4)
1T4	+(int+int)$	r2 (g2)
1E2	+(int+int)$	s6
1E2+6	(int+int)$	s7
1E2+6(7	int+int)$	s11
1E2+6(7int11	+int)$	r4 (g10)
1E2+6(7T10	+int)$	r2 (g8)
1E2+6(7E8	+int)$	s12
1E2+6(7E8+12	int)$	s11
1E2+6(7E8+12int11	)$	r4 (g13)
1E2+6(7E8+12T13	)$	

	int	(	)	+	$	T	E
1	s5					s4	s2
2				s6	ACCEPT		
3				r3	r3		
4				r2	r2		
5				r4	r4		
6	s5	s7				s3	
7	s11	s14				s10	s8
8			s9	s12			
9				r5	r5		
10			r2	r2			
11			r4	r4			
12	s11					s13	
13			r3	r3			
14	s11		s14			s10	s15
15			s16	s12			
16			r5	r5			

- Input: int+(int+int)

$S \rightarrow E$ (1)
$E \rightarrow T$ (2)
$E \rightarrow E + T$ (3)
$T \rightarrow \texttt{int}$ (4)
$T \rightarrow (E)$ (5)

LR(1) Parser

Stack	Input	Action
1	int+(int+int)$	s5
1int5	+(int+int)$	r4 (g4)
1T4	+(int+int)$	r2 (g2)
1E2	+(int+int)$	s6
1E2+6	(int+int)$	s7
1E2+6(7	int+int)$	s11
1E2+6(7int11	+int)$	r4 (g10)
1E2+6(7T10	+int)$	r2 (g8)
1E2+6(7E8	+int)$	s12
1E2+6(7E8+12	int)$	s11
1E2+6(7E8+12int11	)$	r4 (g13)
1E2+6(7E8+12T13	)$	r3
1E2+6(7E	)$	

	int	(	)	+	$	T	E
1	s5					s4	s2
2				s6	ACCEPT		
3				r3	r3		
4				r2	r2		
5				r4	r4		
6	s5	s7				s3	
7	s11	s14				s10	s8
8			s9	s12			
9				r5	r5		
10			r2	r2			
11			r4	r4			
12	s11					s13	
13			r3	r3			
14	s11		s14			s10	s15
15			s16	s12			
16			r5	r5			

- Input: int+(int+int)

$S \rightarrow E$ (1)
$E \rightarrow T$ (2)
$E \rightarrow E + T$ (3)
$T \rightarrow \texttt{int}$ (4)
$T \rightarrow (E)$ (5)

LR(1) Parser

Stack	Input	Action
1	int+(int+int)$	s5
1int5	+(int+int)$	r4 (g4)
1T4	+(int+int)$	r2 (g2)
1E2	+(int+int)$	s6
1E2+6	(int+int)$	s7
1E2+6(7	int+int)$	s11
1E2+6(7int11	+int)$	r4 (g10)
1E2+6(7T10	+int)$	r2 (g8)
1E2+6(7E8	+int)$	s12
1E2+6(7E8+12	int)$	s11
1E2+6(7E8+12int11	)$	r4 (g13)
1E2+6(7E8+12T13	)$	r3 (g8)
1E2+6(7E8	)$	

	int	(	)	+	$	T	E
1	s5					s4	s2
2				s6	ACCEPT		
3				r3	r3		
4				r2	r2		
5				r4	r4		
6	s5	s7				s3	
7	s11	s14				s10	s8
8			s9	s12			
9				r5	r5		
10			r2	r2			
11			r4	r4			
12	s11					s13	
13			r3	r3			
14	s11		s14			s10	s15
15			s16	s12			
16			r5	r5			

- Input: int+(int+int)

$S \rightarrow E$ (1)
$E \rightarrow T$ (2)
$E \rightarrow E + T$ (3)
$T \rightarrow \texttt{int}$ (4)
$T \rightarrow (E)$ (5)

LR(1) Parser

Stack	Input	Action
1	int+(int+int)$	s5
1int5	+(int+int)$	r4 (g4)
1T4	+(int+int)$	r2 (g2)
1E2	+(int+int)$	s6
1E2+6	(int+int)$	s7
1E2+6(7	int+int)$	s11
1E2+6(7int11	+int)$	r4 (g10)
1E2+6(7T10	+int)$	r2 (g8)
1E2+6(7E8	+int)$	s12
1E2+6(7E8+12	int)$	s11
1E2+6(7E8+12int11	)$	r4 (g13)
1E2+6(7E8+12T13	)$	r3 (g8)
1E2+6(7E8	)$	s9
1E2+6(7E8)9	$	

	int	(	)	+	$	T	E
1	s5					s4	s2
2				s6	ACCEPT		
3				r3	r3		
4				r2	r2		
5				r4	r4		
6	s5	s7				s3	
7	s11	s14				s10	s8
8			s9	s12			
9				r5	r5		
10			r2	r2			
11			r4	r4			
12	s11					s13	
13			r3	r3			
14	s11		s14			s10	s15
15			s16	s12			
16			r5	r5			

- Input: int+(int+int)

$S \rightarrow E$ (1)
$E \rightarrow T$ (2)
$E \rightarrow E + T$ (3)
$T \rightarrow \texttt{int}$ (4)
$T \rightarrow (E)$ (5)

LR(1) Parser

Stack	Input	Action
1	int+(int+int)$	s5
1int5	+(int+int)$	r4 (g4)
1T4	+(int+int)$	r2 (g2)
1E2	+(int+int)$	s6
1E2+6	(int+int)$	s7
1E2+6(7	int+int)$	s11
1E2+6(7int11	+int)$	r4 (g10)
1E2+6(7T10	+int)$	r2 (g8)
1E2+6(7E8	+int)$	s12
1E2+6(7E8+12	int)$	s11
1E2+6(7E8+12int11	)$	r4 (g13)
1E2+6(7E8+12T13	)$	r3 (g8)
1E2+6(7E8	)$	s9
1E2+6(7E8)9	$	r5
1E2+6T	$	

	int	(	)	+	$	T	E
1	s5					s4	s2
2				s6	ACCEPT		
3				r3	r3		
4				r2	r2		
5				r4	r4		
6	s5	s7				s3	
7	s11	s14				s10	s8
8			s9	s12			
9				r5	r5		
10			r2	r2			
11			r4	r4			
12	s11					s13	
13			r3	r3			
14	s11		s14			s10	s15
15			s16	s12			
16			r5	r5			

- Input: int+(int+int)

$S \rightarrow E$ (1)
$E \rightarrow T$ (2)
$E \rightarrow E + T$ (3)
$T \rightarrow \texttt{int}$ (4)
$T \rightarrow (E)$ (5)

LR(1) Parser

Stack	Input	Action
1	int+(int+int)$	s5
1int5	+(int+int)$	r4 (g4)
1T4	+(int+int)$	r2 (g2)
1E2	+(int+int)$	s6
1E2+6	(int+int)$	s7
1E2+6(7	int+int)$	s11
1E2+6(7int11	+int)$	r4 (g10)
1E2+6(7T10	+int)$	r2 (g8)
1E2+6(7E8	+int)$	s12
1E2+6(7E8+12	int)$	s11
1E2+6(7E8+12int11	)$	r4 (g13)
1E2+6(7E8+12T13	)$	r3 (g8)
1E2+6(7E8	)$	s9
1E2+6(7E8)9	$	r5 (g3)
1E2+6T3	$	

	int	(	)	+	$	T	E
1	s5					s4	s2
2				s6	ACCEPT		
3				r3	r3		
4				r2	r2		
5				r4	r4		
6	s5	s7				s3	
7	s11	s14				s10	s8
8			s9	s12			
9				r5	r5		
10			r2	r2			
11			r4	r4			
12	s11					s13	
13			r3	r3			
14	s11		s14			s10	s15
15			s16	s12			
16			r5	r5			

- Input: int+(int+int)

$S \rightarrow E$ (1)
$E \rightarrow T$ (2)
$E \rightarrow E + T$ (3)
$T \rightarrow \texttt{int}$ (4)
$T \rightarrow (E)$ (5)

LR(1) Parser

Stack	Input	Action
1	int+(int+int)$	s5
1int5	+(int+int)$	r4 (g4)
1T4	+(int+int)$	r2 (g2)
1E2	+(int+int)$	s6
1E2+6	(int+int)$	s7
1E2+6(7	int+int)$	s11
1E2+6(7int11	+int)$	r4 (g10)
1E2+6(7T10	+int)$	r2 (g8)
1E2+6(7E8	+int)$	s12
1E2+6(7E8+12	int)$	s11
1E2+6(7E8+12int11	)$	r4 (g13)
1E2+6(7E8+12T13	)$	r3 (g8)
1E2+6(7E8	)$	s9
1E2+6(7E8)9	$	r5 (g3)
1E2+6T3	$	r3
1E	$	

	int	(	)	+	$	T	E
1	s5					s4	s2
2				s6	ACCEPT		
3				r3	r3		
4				r2	r2		
5				r4	r4		
6	s5	s7				s3	
7	s11	s14				s10	s8
8			s9	s12			
9				r5	r5		
10			r2	r2			
11			r4	r4			
12	s11					s13	
13			r3	r3			
14	s11		s14			s10	s15
15			s16	s12			
16			r5	r5			

- Input: int+(int+int)

$S \rightarrow E$ (1)
$E \rightarrow T$ (2)
$E \rightarrow E + T$ (3)
$T \rightarrow \texttt{int}$ (4)
$T \rightarrow (E)$ (5)

LR(1) Parser

Stack	Input	Action
1	int+(int+int)$	s5
1int5	+(int+int)$	r4 (g4)
1T4	+(int+int)$	r2 (g2)
1E2	+(int+int)$	s6
1E2+6	(int+int)$	s7
1E2+6(7	int+int)$	s11
1E2+6(7int11	+int)$	r4 (g10)
1E2+6(7T10	+int)$	r2 (g8)
1E2+6(7E8	+int)$	s12
1E2+6(7E8+12	int)$	s11
1E2+6(7E8+12int11	)$	r4 (g13)
1E2+6(7E8+12T13	)$	r3 (g8)
1E2+6(7E8	)$	s9
1E2+6(7E8)9	$	r5 (g3)
1E2+6T3	$	r3 (g2)
1E2	$	

	int	(	)	+	$	T	E
1	s5					s4	s2
2				s6	ACCEPT		
3				r3	r3		
4				r2	r2		
5				r4	r4		
6	s5	s7				s3	
7	s11	s14				s10	s8
8			s9	s12			
9				r5	r5		
10			r2	r2			
11			r4	r4			
12	s11					s13	
13			r3	r3			
14	s11		s14			s10	s15
15			s16	s12			
16			r5	r5			

- Input: int+(int+int)

$S \rightarrow E$ (1)
$E \rightarrow T$ (2)
$E \rightarrow E + T$ (3)
$T \rightarrow \texttt{int}$ (4)
$T \rightarrow (E)$ (5)

LR(1) Parser

Stack	Input	Action
1	int+(int+int)$	s5
1int5	+(int+int)$	r4 (g4)
1T4	+(int+int)$	r2 (g2)
1E2	+(int+int)$	s6
1E2+6	(int+int)$	s7
1E2+6(7	int+int)$	s11
1E2+6(7int11	+int)$	r4 (g10)
1E2+6(7T10	+int)$	r2 (g8)
1E2+6(7E8	+int)$	s12
1E2+6(7E8+12	int)$	s11
1E2+6(7E8+12int11	)$	r4 (g13)
1E2+6(7E8+12T13	)$	r3 (g8)
1E2+6(7E8	)$	s9
1E2+6(7E8)9	$	r5 (g3)
1E2+6T3	$	r3 (g2)
1E2	$	Accept

	int	(	)	+	$	T	E
1	s5					s4	s2
2				s6	ACCEPT		
3				r3	r3		
4				r2	r2		
5				r4	r4		
6	s5	s7				s3	
7	s11	s14				s10	s8
8			s9	s12			
9				r5	r5		
10			r2	r2			
11			r4	r4			
12	s11					s13	
13			r3	r3			
14	s11		s14			s10	s15
15			s16	s12			
16			r5	r5			

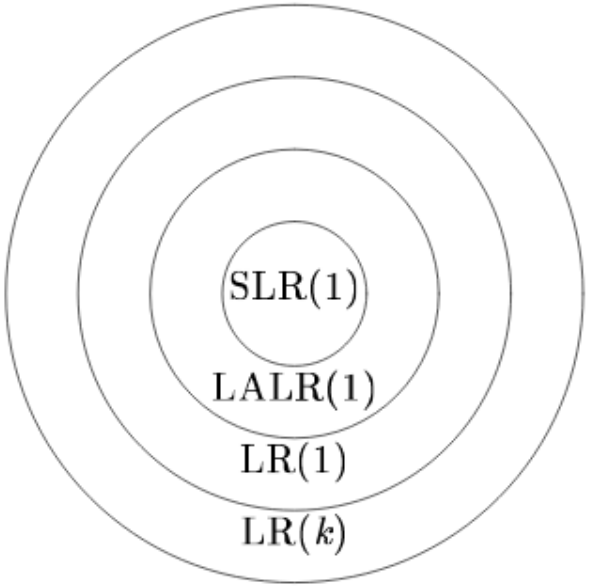
- Input: int+(int+int)

$S \rightarrow E$ (1)
$E \rightarrow T$ (2)
$E \rightarrow E + T$ (3)
$T \rightarrow \texttt{int}$ (4)
$T \rightarrow (E)$ (5)

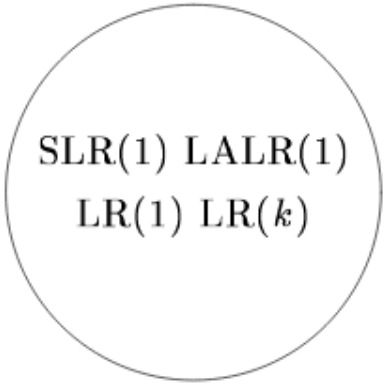
The Power of LR(1)

- Any LR(0) grammar is LR(1).
- Any LL(1) grammar is LR(1).
- Any deterministic CFL (a CFL parseable by a deterministic pushdown automaton) has an LR(1) grammar.
- Any LL(k) language is LR(1), though individual LL(k) grammars might not be.
- Any LR(k) language is LR(1), though individual LR(k) grammars might not be.

The Power of LR(1)



Grammars



Languages

LR(1) Automata are Huge

- In a grammar with n terminals, could in theory be n times as large as the LR(0) automaton.
  - Replicate each state with all possible lookaheads.
- LR(1) tables for practical programming languages can have hundreds of thousands or even millions of states.
- Consequently, LR(1) parsers are rarely used in practice.

LR(1) Automata are Huge

Is there a way to get the power of LR(1) without the huge table size?

Why is LR(1) powerful?

- Intuitively, for two reasons:
- Lookahead makes handle-finding easier.
  - The LR(1) automaton can predict whether it needs to reduce based on more information.
  - The LR(0) automaton says whether there could be a handle later on based on no right context.
- More states encode more information.
  - LR(1) lookaheads are very good because there's a greater number of states to be in.
- Goal: Incorporate lookahead without increasing the number of states.

Revisiting Shift/Reduce Conflicts

- A shift/reduce conflict is a state that looks like this:

  $A \rightarrow \omega \cdot$

  $B \rightarrow \alpha \cdot \beta$

- In LR(0), this is simply not allowed.
- In LR(1), this can be avoided by using lookahead to determine whether to shift or reduce.
- Can we get some of the lookahead power of LR(1) without the huge tables?

Outline

- Introduction
- Shift-reduce Parsing
- **LR Parsing**
  - LR(0)
  - LR(1)
  - **SLR**
  - LALR
- Ambiguity
- Error-handling