Introduction to Simple LR Parsing

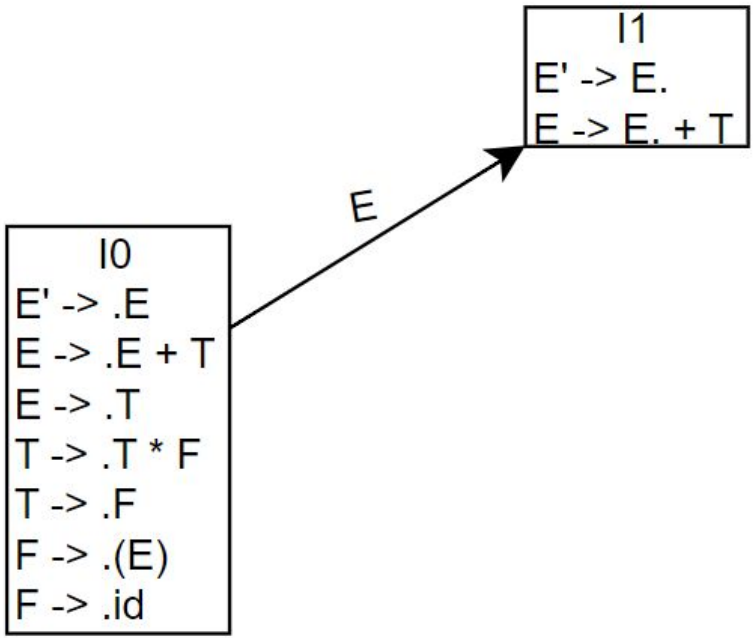
Prepared By: Sukarna Sarker

In LR parsing:

1. L means Left to right scanning and R means rightmost derivation
2. k for the number of input symbols of lookahead that are used in making parsing decisions
3. When (k) is omitted, k is assumed to be 1

LR(0) automaton for the expression grammar(Step by step)

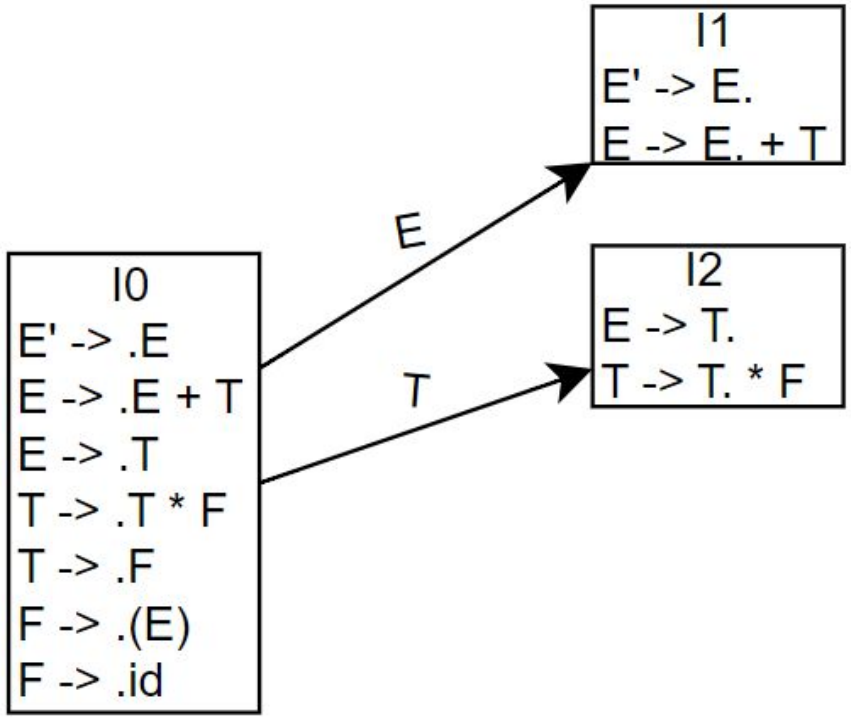
E -> E’
E -> E + T | T
T -> T * F | F
F -> (E) | id



Prepared By: Sukarna Sarker

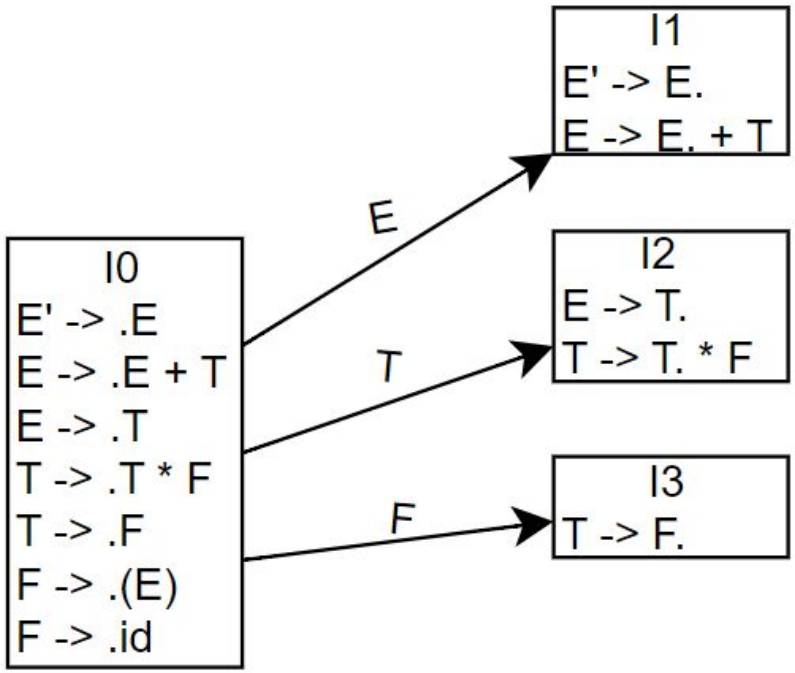
LR(0) automaton for the expression grammar(Step by step)

E -> E’
E -> E + T | T
T -> T * F | F
F -> (E) | id



LR(0) automaton for the expression grammar(Step by step)

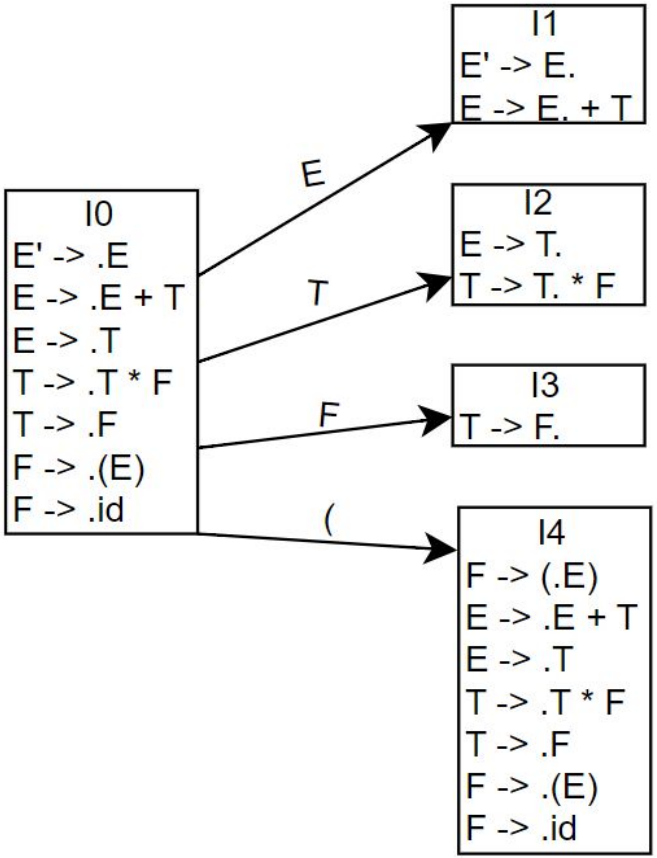
E -> E’
E -> E + T | T
T -> T * F | F
F -> (E) | id



Prepared By: Sukarna Sarker

LR(0) automaton for the expression grammar(Step by step)

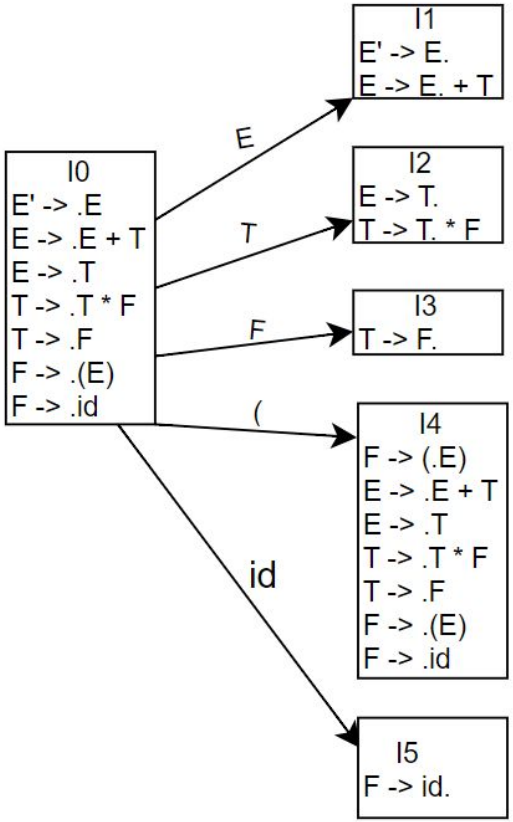
E -> E’
E -> E + T | T
T -> T * F | F
F -> (E) | id



Prepared By: Sukarna Sarker

LR(0) automaton for the expression grammar(Step by step)

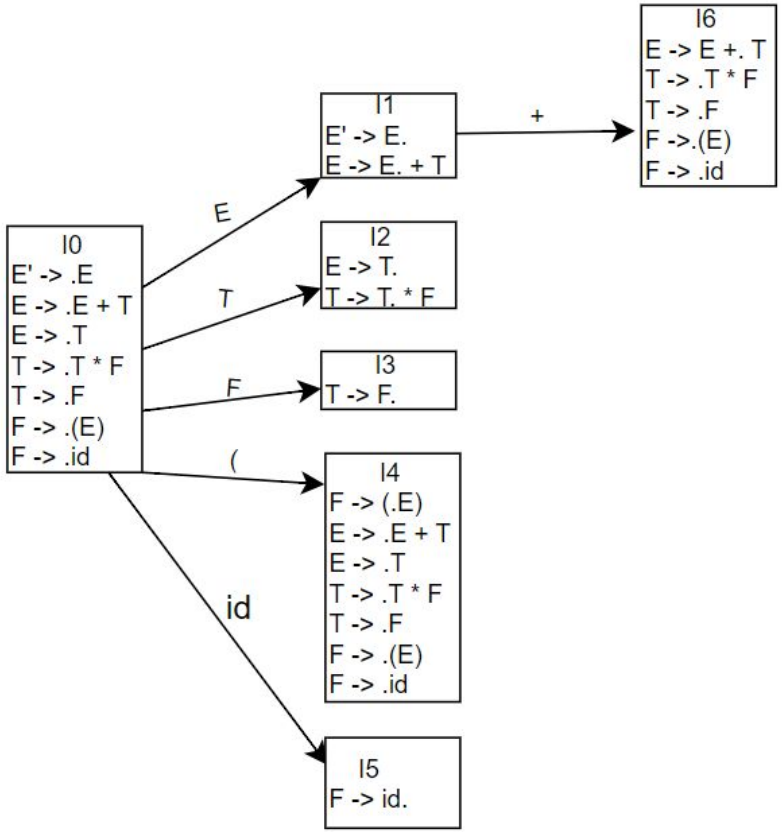
E -> E’
E -> E + T | T
T -> T * F | F
F -> (E) | id



Prepared By: Sukarna Sarker

LR(0) automaton for the expression grammar(Step by step)

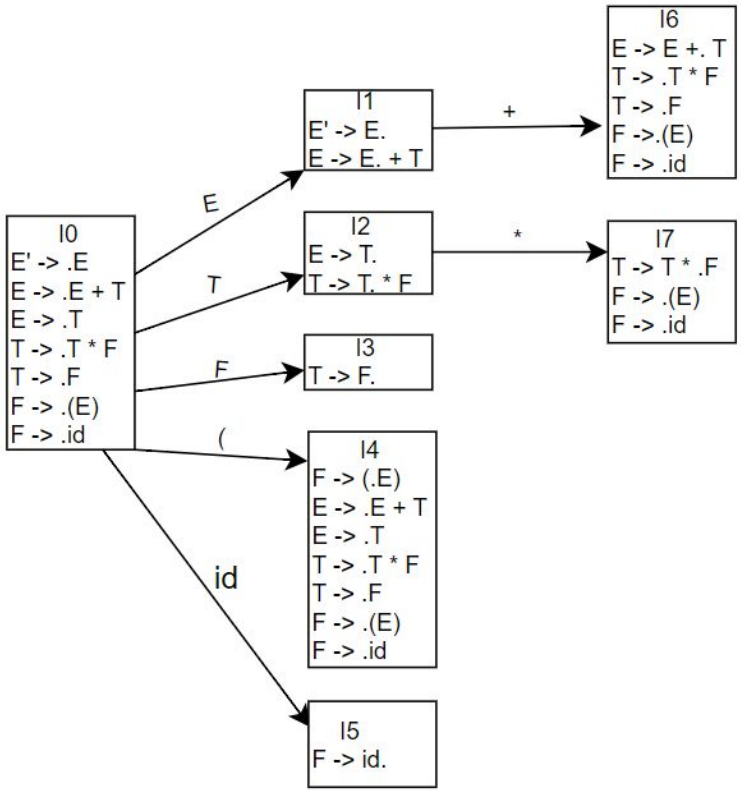
E -> E’
E -> E + T | T
T -> T * F | F
F -> (E) | id



Prepared By: Sukarna Sarker

LR(0) automaton for the expression grammar(Step by step)

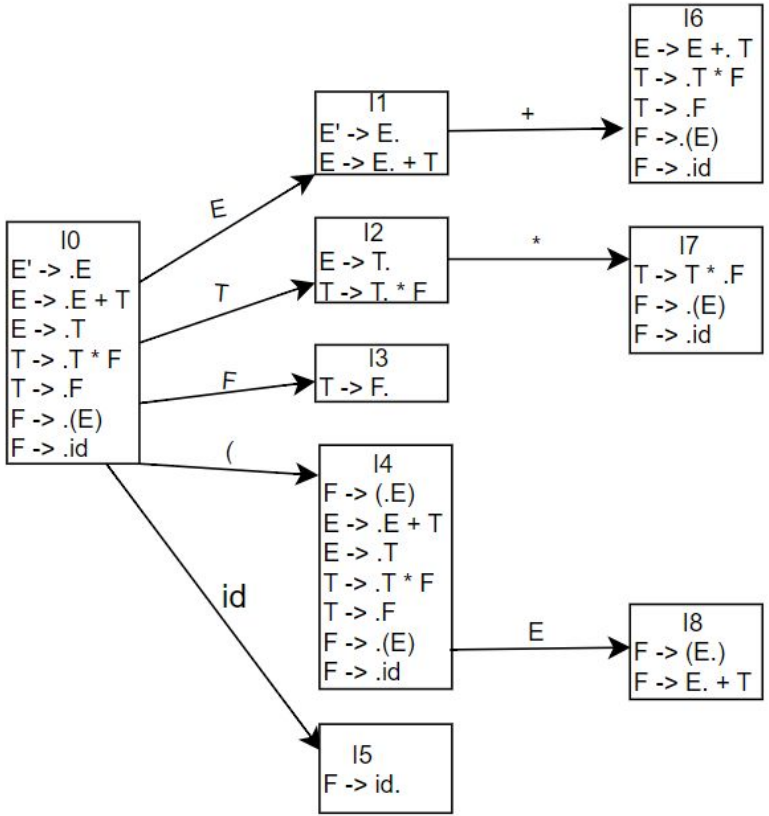
E -> E’
E -> E + T | T
T -> T * F | F
F -> (E) | id



Prepared By: Sukarna Sarker

LR(0) automaton for the expression grammar(Step by step)

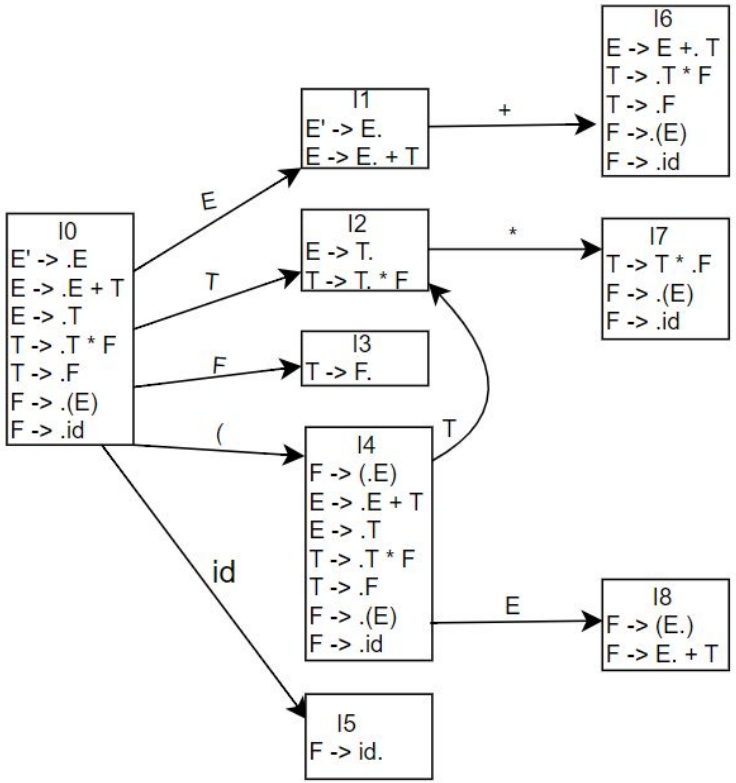
E -> E’
E -> E + T | T
T -> T * F | F
F -> (E) | id



Prepared By: Sukarna Sarker

LR(0) automaton for the expression grammar(Step by step)

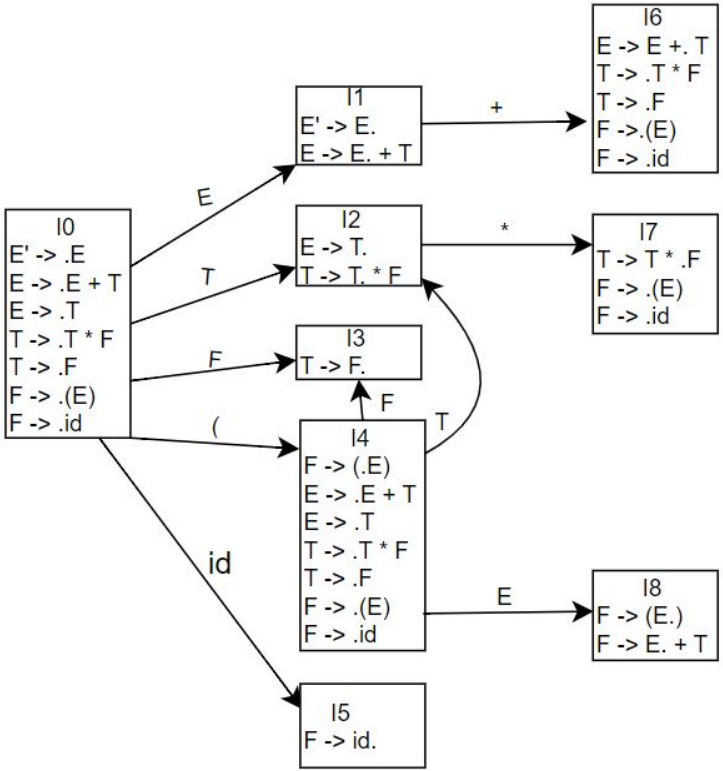
E -> E’
E -> E + T | T
T -> T * F | F
F -> (E) | id



Prepared By: Sukarna Sarker

LR(0) automaton for the expression grammar(Step by step)

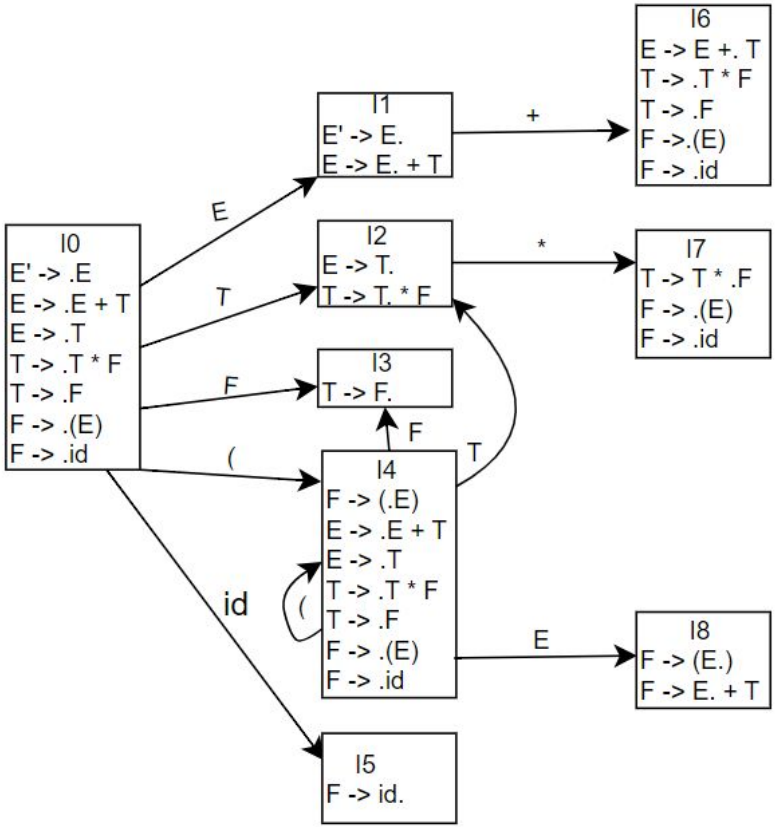
E -> E’
E -> E + T | T
T -> T * F | F
F -> (E) | id



Prepared By: Sukarna Sarker

LR(0) automaton for the expression grammar(Step by step)

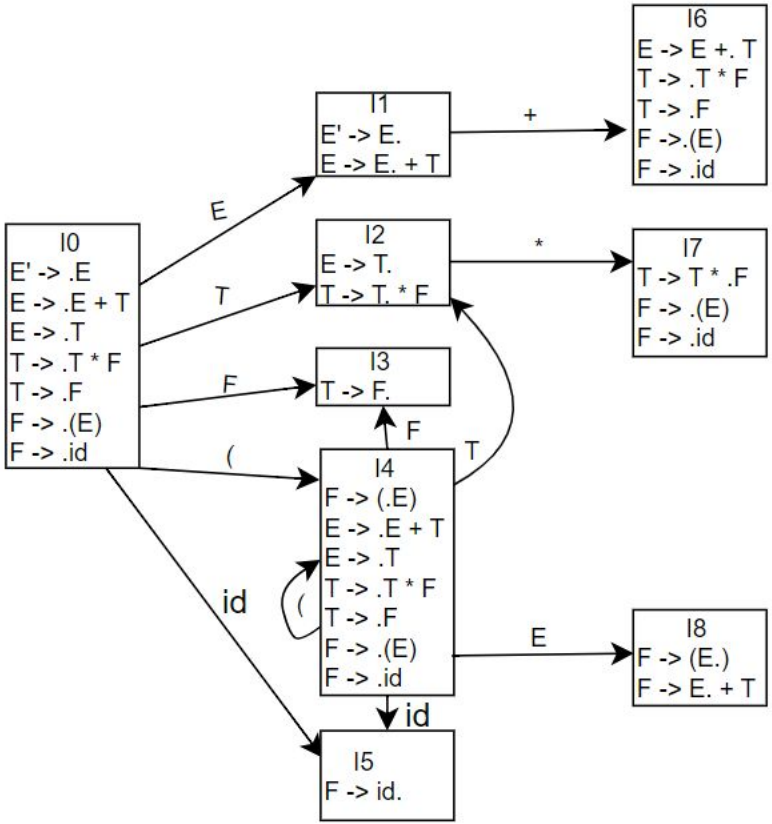
E -> E’
E -> E + T | T
T -> T * F | F
F -> (E) | id



Prepared By: Sukarna Sarker

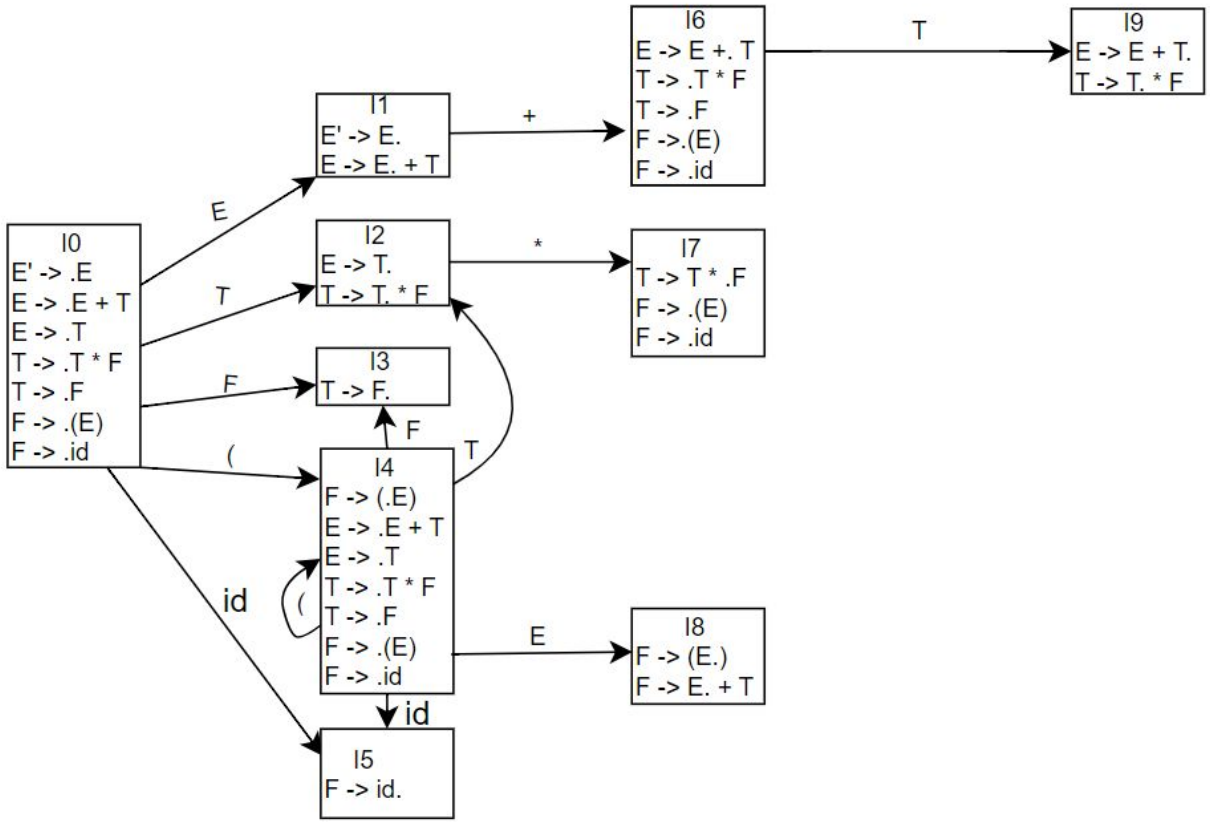
LR(0) automaton for the expression grammar(Step by step)

E -> E’
E -> E + T | T
T -> T * F | F
F -> (E) | id



LR(0) automaton for the expression grammar(Step by step)

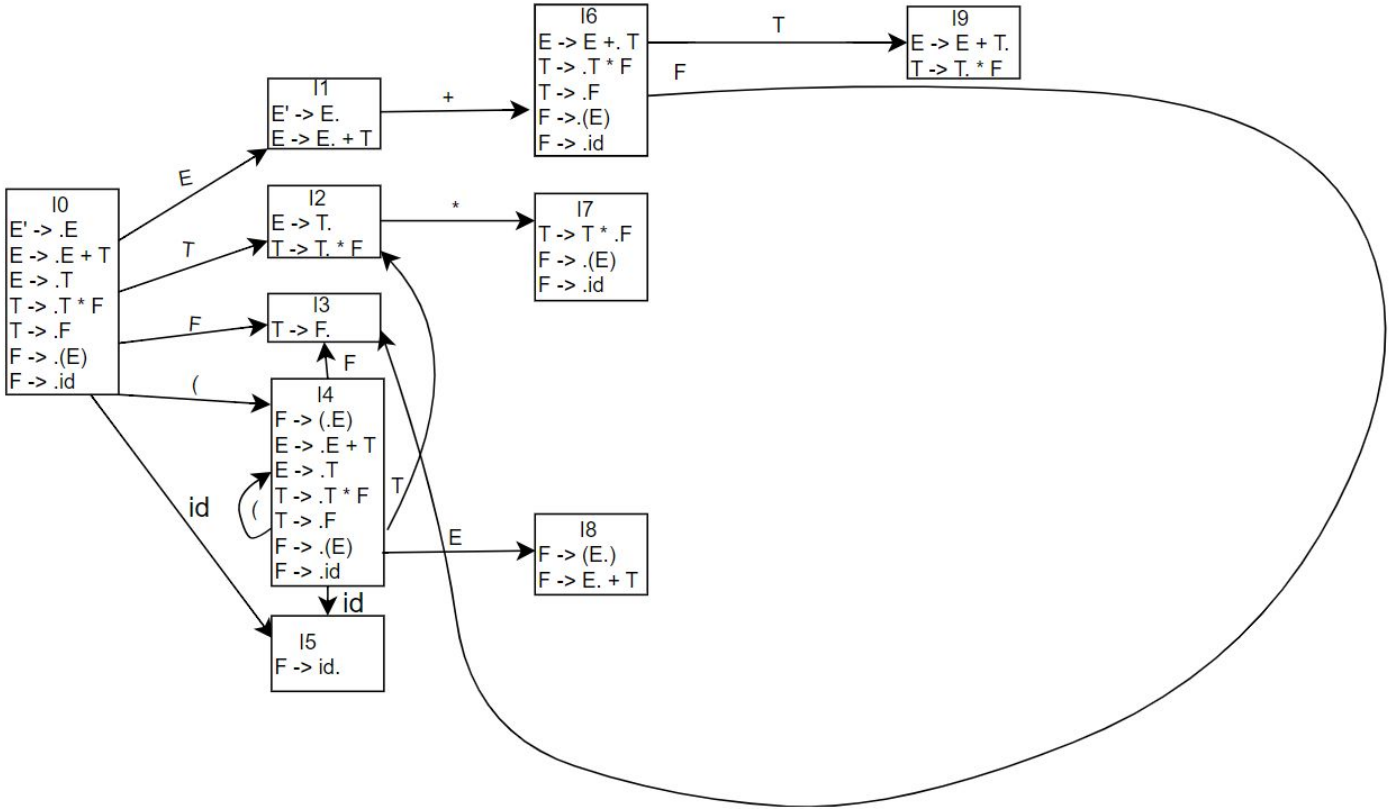
E -> E’
E -> E + T | T
T -> T * F | F
F -> (E) | id



Prepared By: Sukarna Sarker

LR(0) automaton for the expression grammar(Step by step)

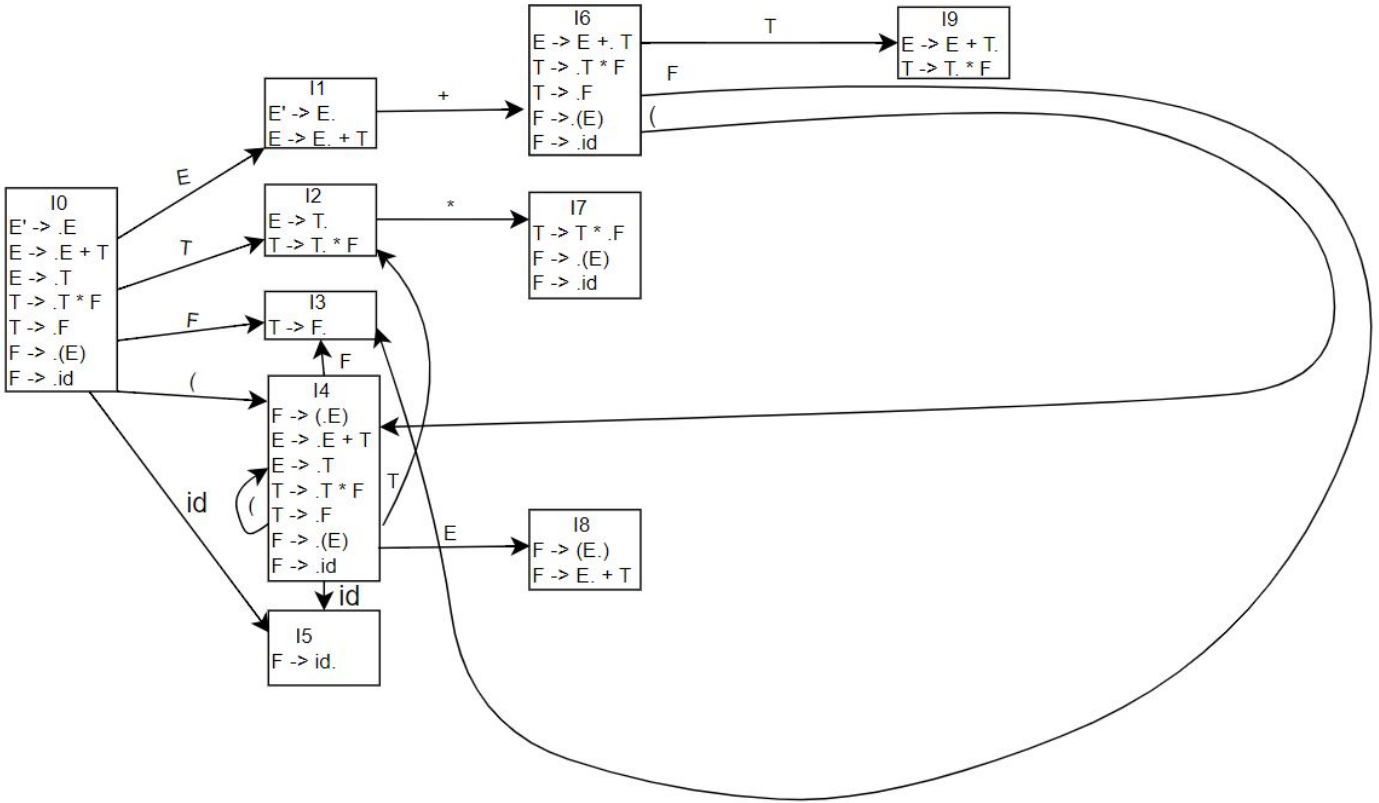
E -> E’
E -> E + T | T
T -> T * F | F
F -> (E) | id



Prepared By: Sukarna Sarker

LR(0) automaton for the expression grammar(Step by step)

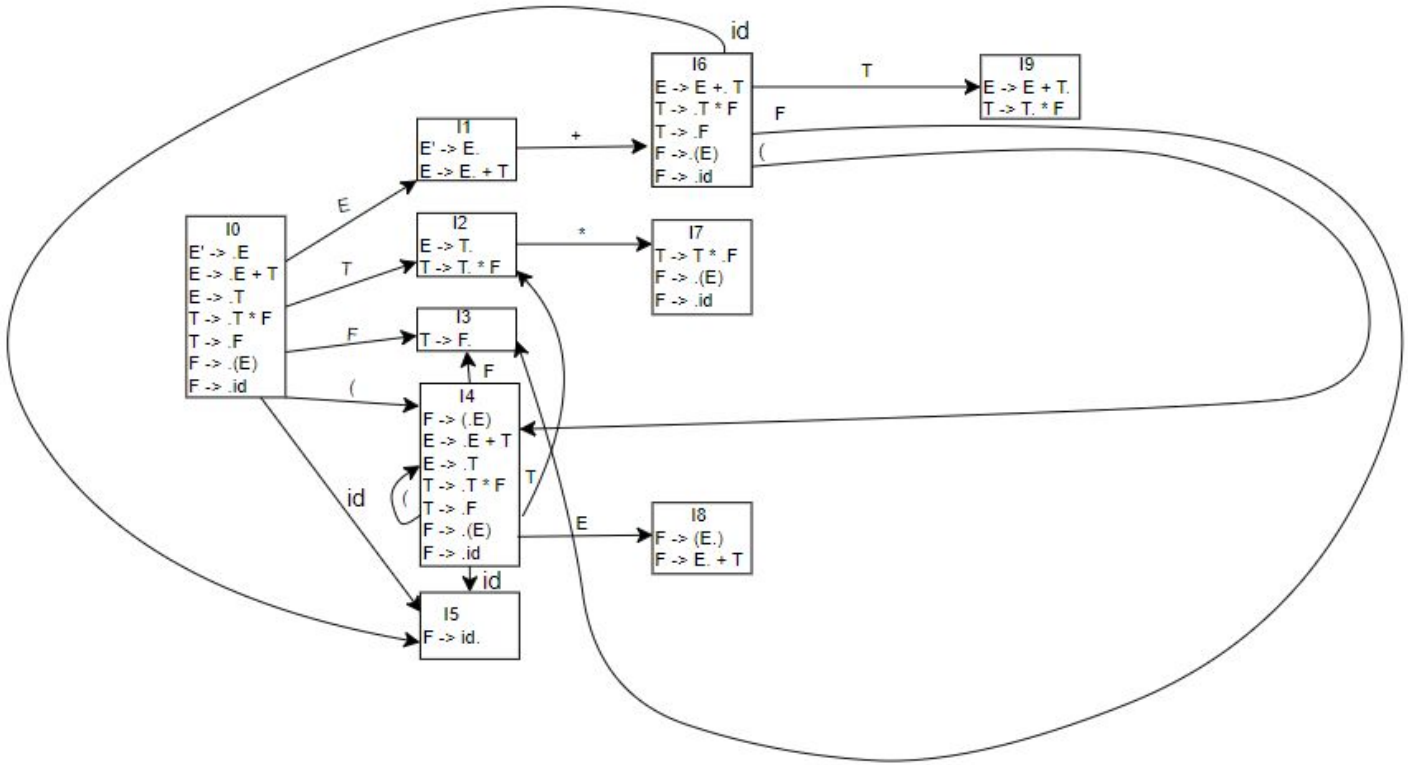
E -> E’
E -> E + T | T
T -> T * F | F
F -> (E) | id



Prepared By: Sukarna Sarker

LR(0) automaton for the expression grammar(Step by step)

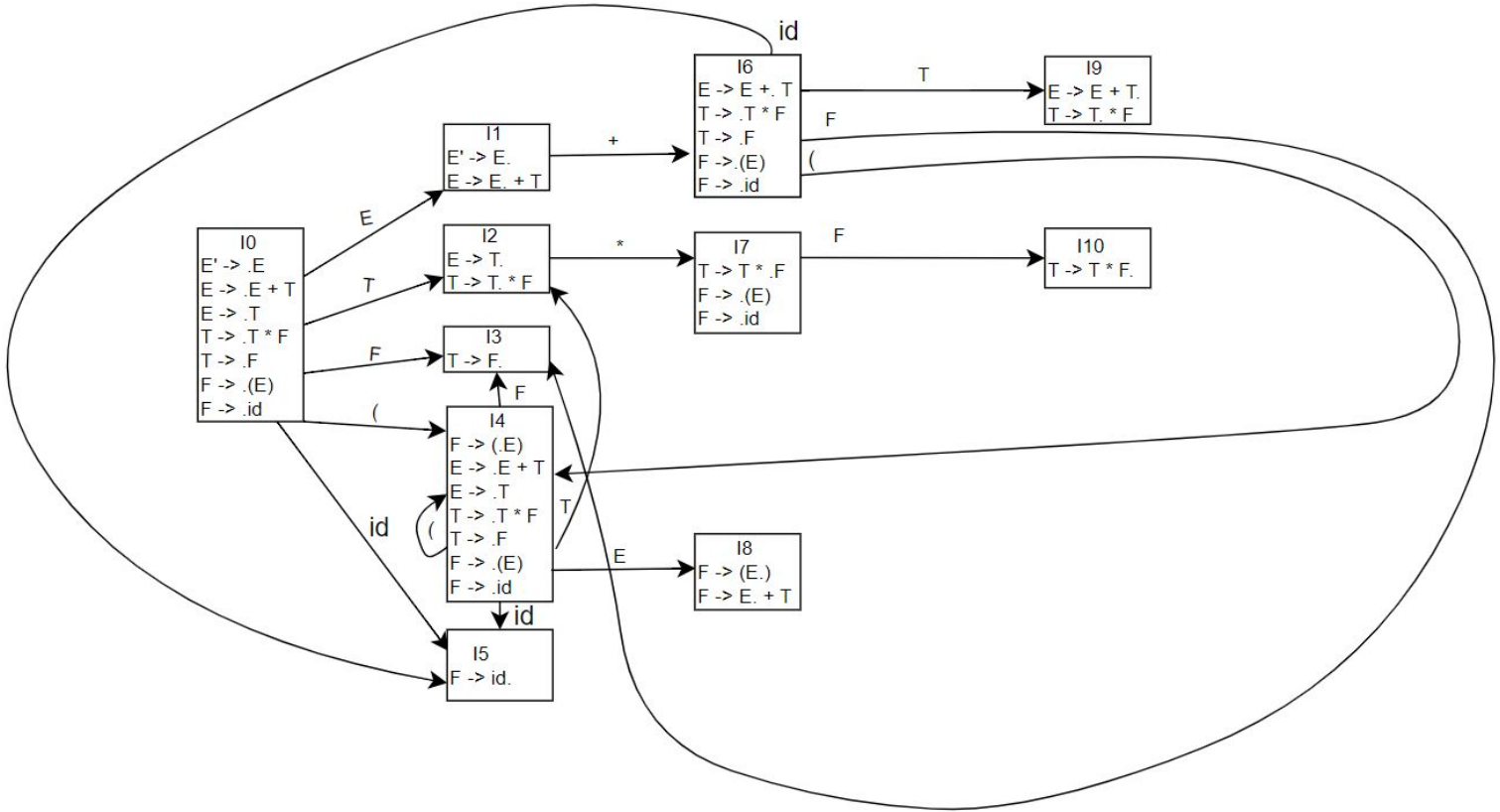
E -> E’
E -> E + T | T
T -> T * F | F
F -> (E) | id



Prepared By: Sukarna Sarker

LR(0) automaton for the expression grammar(Step by step)

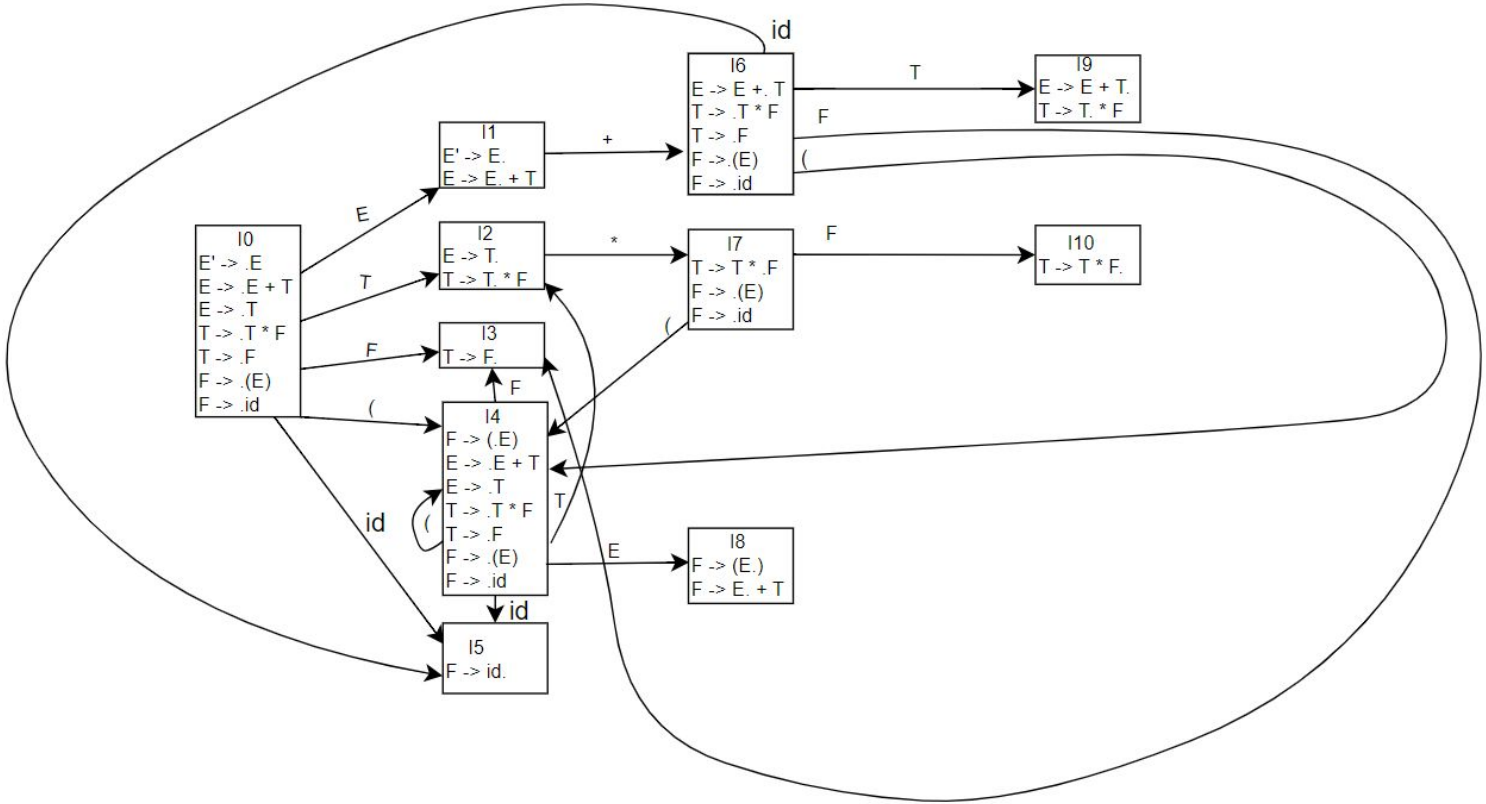
E -> E’
E -> E + T | T
T -> T * F | F
F -> (E) | id



Prepared By: Sukarna Sarker

LR(0) automaton for the expression grammar(Step by step)

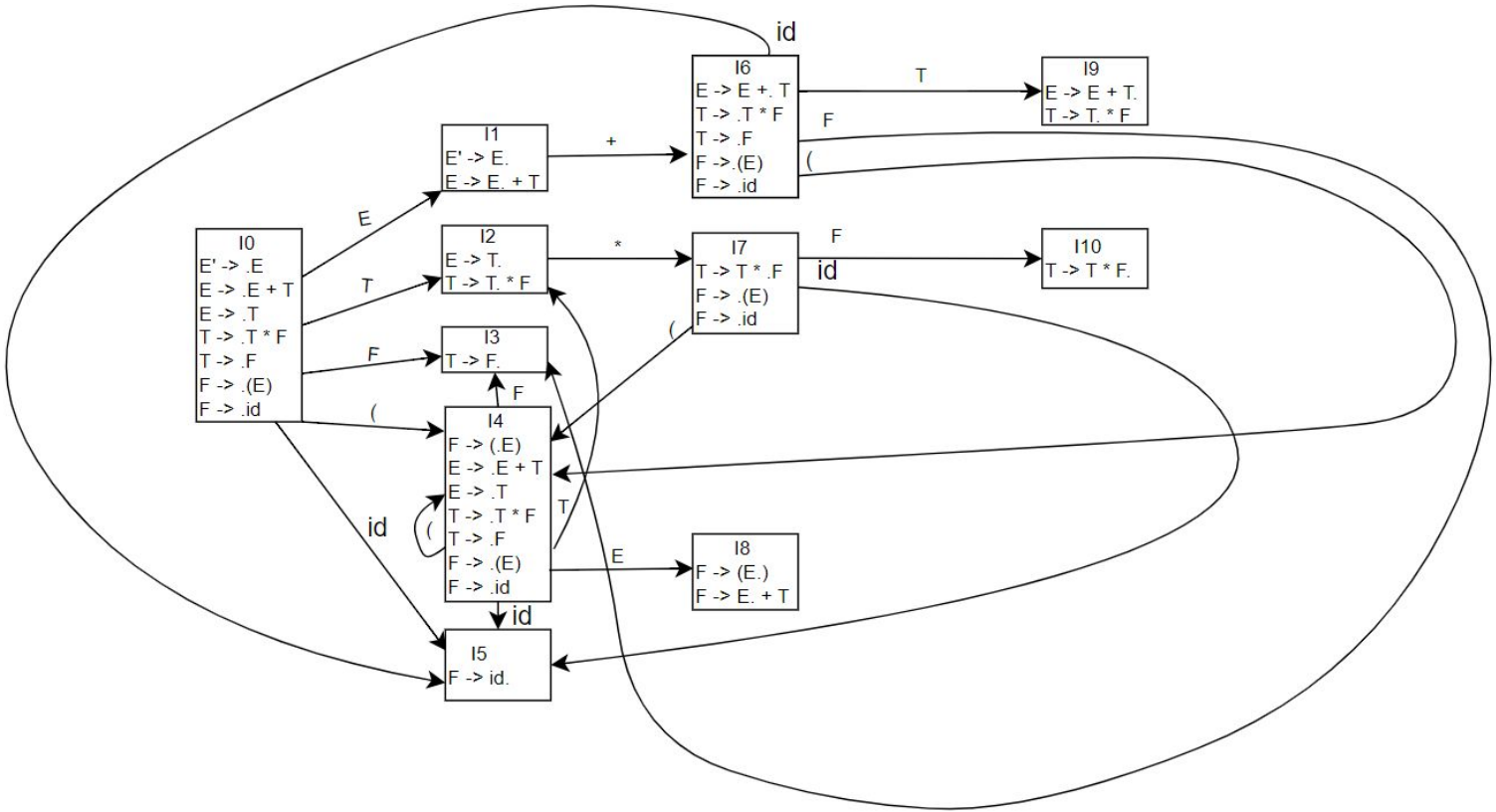
E -> E’
E -> E + T | T
T -> T * F | F
F -> (E) | id



Prepared By: Sukarna Sarker

LR(0) automaton for the expression grammar(Step by step)

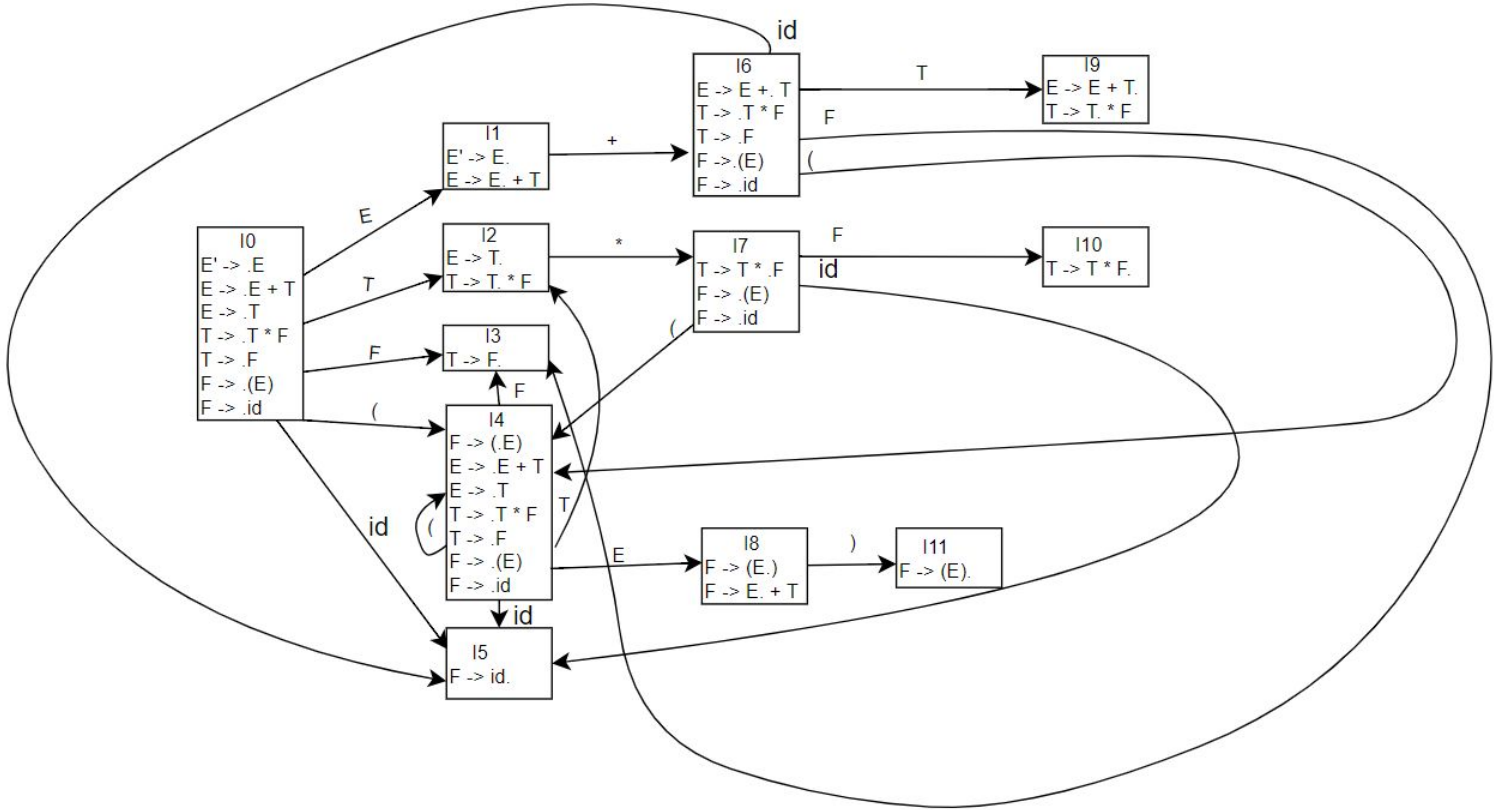
E -> E’
E -> E + T | T
T -> T * F | F
F -> (E) | id



Prepared By: Sukarna Sarker

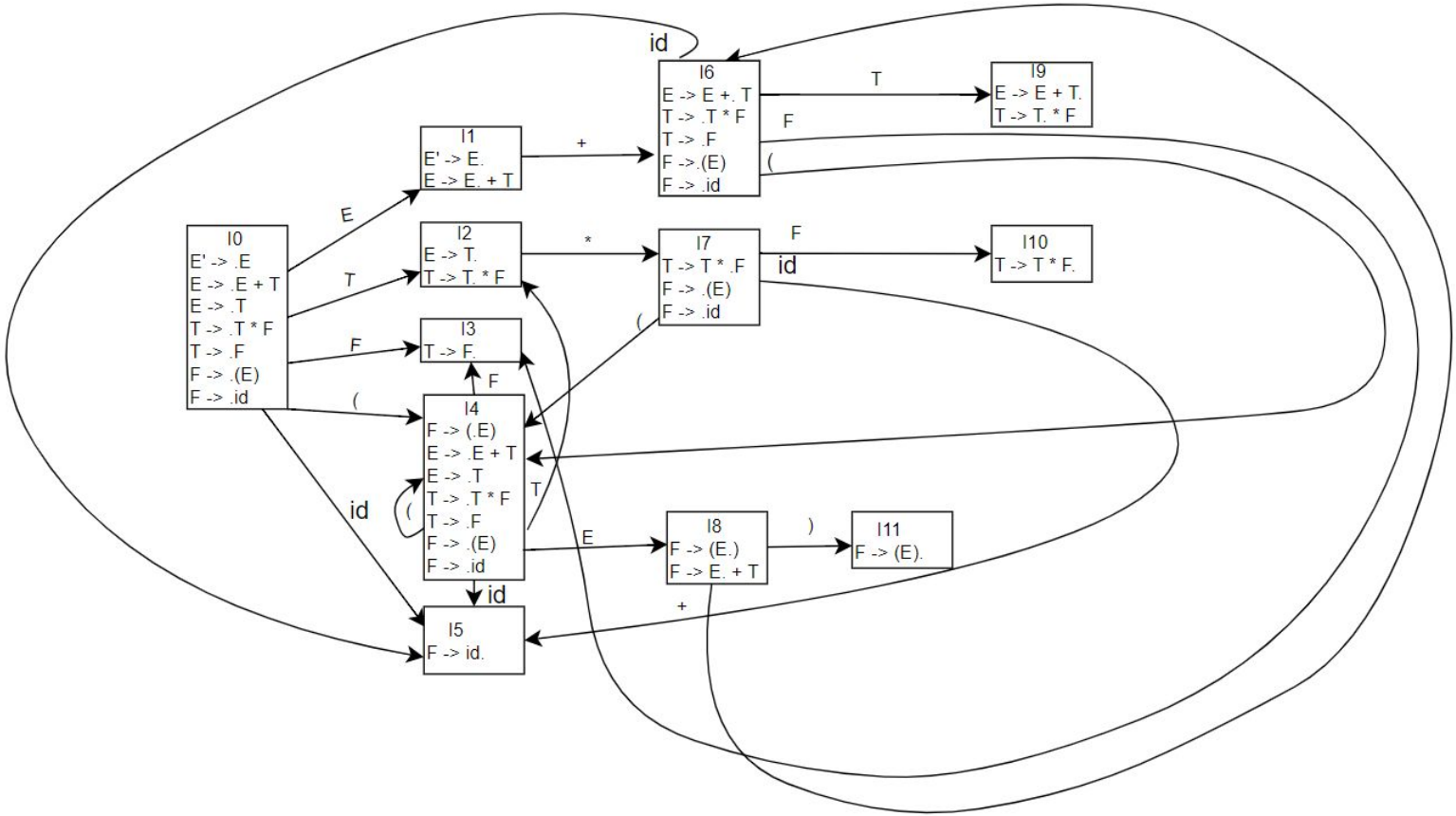
LR(0) automaton for the expression grammar(Step by step)

E -> E’
E -> E + T | T
T -> T * F | F
F -> (E) | id



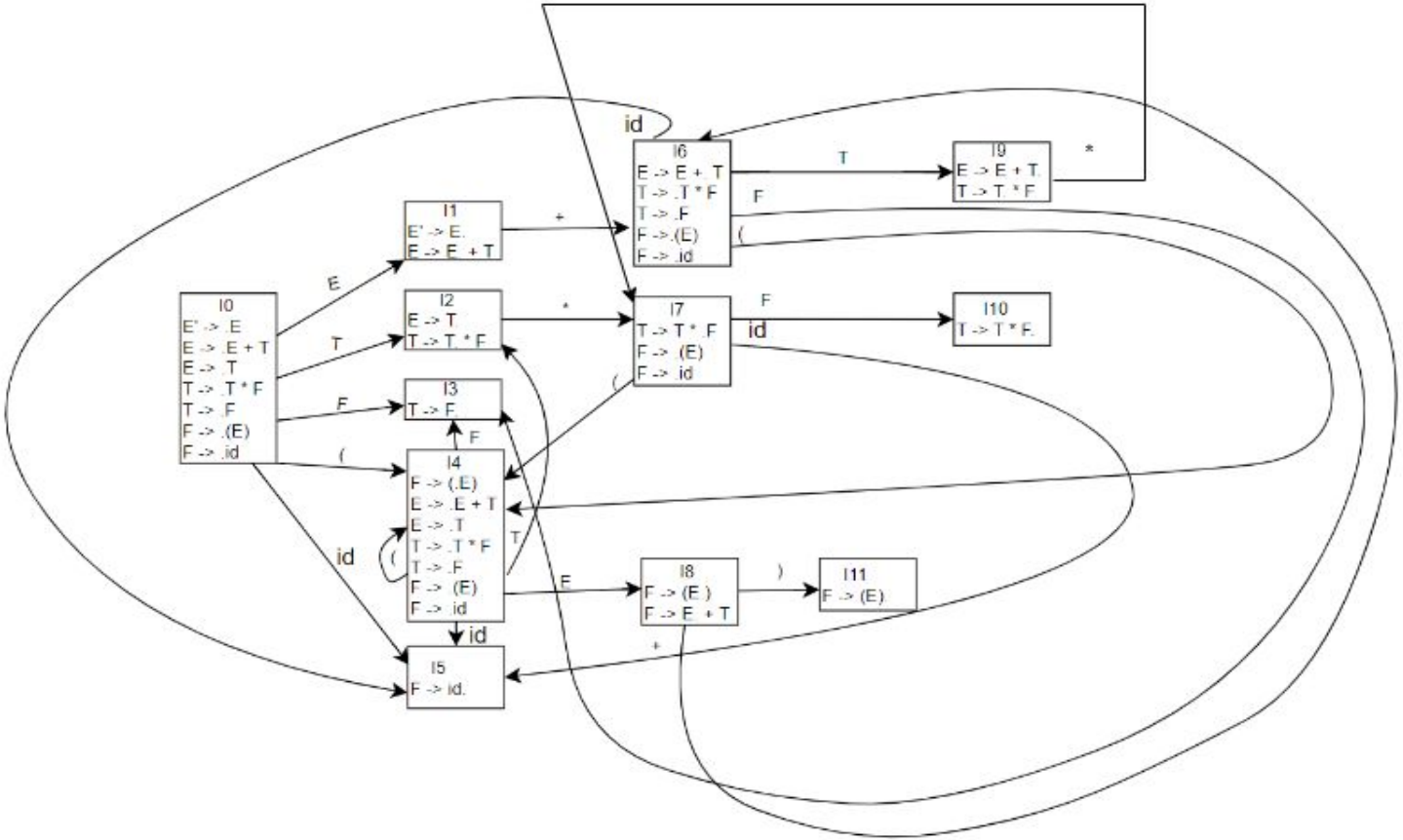
LR(0) automaton for the expression grammar(Step by step)

E -> E’
E -> E + T | T
T -> T * F | F
F -> (E) | id



LR(0) automaton for the expression grammar(Step by step)

E -> E’
E -> E + T | T
T -> T * F | F
F -> (E) | id



Prepared By: Sukarna Sarker

LR(0) Parsing table for expression grammar

(1) E -> E + T
(2) E -> T
(3) T -> T * F
(4) T -> F
(5) F -> (E)
(6) F -> id

State	Action						Go - To		
	id	+	*	(	)	$	E	T	F
0	S5			S4			1	2	3
1		S6				Accept			
2	R2	R2	S7 R2	R2	R2	R2			
3	R4	R4	R4	R4	R4	R4			
4	S5			S4			8	2	3
5	R6	R6	R6	R6	R6	R6			
6	S5			S4				9	3
7	S5			S4					10
8		S6			S11				
9	R1	R1	R1 S7	R1	R1	R1			
10	R3	R3	R3	R3	R3	R3			
11	R5	R5	R5	R5	R5	R5			

Prepared By: Sukarna Sarker

First and Follow

Line	First	Follow
E -> E + T	First(E) = { (, id }	Follow(E) = { $, + ,) }
E -> T		
T -> T * F	First(T) = { (, id}	Follow(T) = { * , $, + ,) }
T -> F		
F -> (E)	First(F) = { (, id}	Follow(F) = { * , $, + ,) }
F -> id		

Prepared By: Sukarna Sarker

SLR(1) Parsing table for expression grammar

(1) E -> E + T
(2) E -> T
(3) T -> T * F
(4) T -> F
(5) F -> (E)
(6) F -> id

State	Action						Go - To		
	id	+	*	(	)	$	E	T	F
0	S5			S4			1	2	3
1		S6				Accept			
2		R2	S7		R2	R2			
3		R4	R4		R4	R4			
4	S5			S4			8	2	3
5		R6	R6		R6	R6			
6	S5			S4				9	3
7	S5			S4					10
8		S6			S11				
9		R1	S7		R1	R1			
10		R3	R3		R3	R3			
11		R5	R5		R5	R5			

Prepared By: Sukarna Sarker

Moves of an SLR(1) parser on id * id

(1) E -> E + T
(2) E -> T
(3) T -> T * F
(4) T -> F
(5) F -> (E)
(6) F -> id

Line	Stack	Symbol	Input	Action
1	0	$	id * id$	Shift to 5
2	0 5	$id	* id$	Reduce by F -> id
3	0 3	$F	* id$	Reduce by T -> F
4	0 2	$T	* id$	Shift to 7
5	0 2 7	$T *	id$	Shift to 5
6	0 2 7 5	$T * id	$	Reduce by F ->id
7	0 2 7 10	$T * F	$	Reduce by T -> T * F
8	0 2	$T	$	Reduce by E -> T
9	0 1	$E	$	Accept

Prepared By: Sukarna Sarker