文法と言語
ー文脈自由文法とLR構文解析３ー

和田俊和

資料保存場所

http://vrl.sys.wakayama-u.ac.jp/~twada/syspro/

前回問題はLR(0)構文解析できないケースです.

以下詳細を説明します.

文法

(0)<S>→<E>

(1) <E> →<E>+<B>

(2) <E> →<B>

(3) <B> →<B>*<T>

(4) <B> →<T>

(5) <T> → 0

(6) <T> → 1

アイテム集合

集合0

S →・E

＋E →・E+B

＋E →・B

＋B →・B*T

＋B →・T

＋T → ・0

＋T → ・1

集合1

T → 0・

集合2

T → 1・

集合3

B →T・

集合4

S →E・

E →E・+B

集合5

E →B・

B →B・*T

集合6

E →E+・B

＋B →・B*T

＋B →・T

＋T → ・0

＋T → ・1

集合7

B →B*・T

＋T → ・0

＋T → ・1

集合8

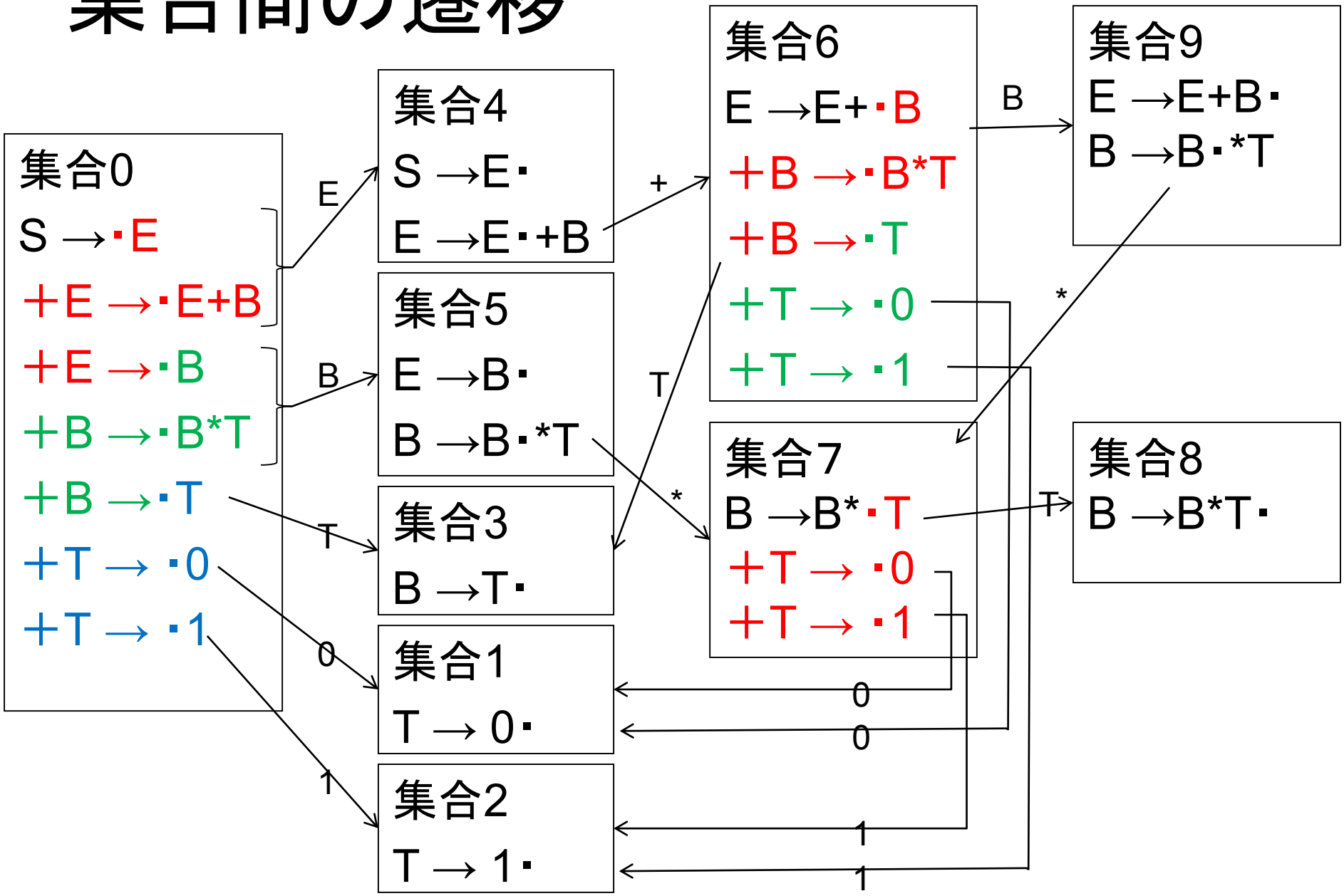
B →B*T・

集合9

E →E+B・

B →B・*T

集合間の遷移


集合0
S →・E
＋E →・E+B
＋E →・B
＋B →・B*T
＋B →・T
＋T → ・0
＋T → ・1
集合4
S →E・
E →E・+B
集合5
E →B・
B →B・*T
集合3
B →T・
集合1
T → 0・
集合2
T → 1・
集合6
E →E+・B
＋B →・B*T
＋B →・T
＋T → ・0
＋T → ・1
集合7
B →B*・T
＋T → ・0
＋T → ・1
集合9
E →E+B・
B →B・*T
集合8
B →B*T・
E
B
T
0
1
+
T
*
B
*
T
0
0
1
1

アイテム集合から状態遷移表

アイテム集合	*	+	0	1	E	B	T
0			1	2	4	5	3
1							
2							
3							
4		6					
5	7						
6			1	2		9	3
7			1	2			8
8							
9	7						

状態遷移表から，アクション・GOTO表

アイテム集合	*	+	0	1	E	B	T
0			1	2	4	5	3
1							
2							
3							
4		6					
5	7						
6			1	2		9	3
7			1	2			8
8							
9	7						

	アクション						GOTO		
状態	*	+	0	1	$		E	B	T
0			s1	s2			4	5	3
1	r5	r5	r5	r5	r5				
2	r6	r6	r6	r6	r6				
3	r4	r4	r4	r4	r4				
4		s6			acc				
5	s7 r2	r2	r2	r2	r2				
6			s1	s2				9	3
7			s1	s2					8
8	r3	r3	r3	r3	r3				
9	s7 r1	r1	r1	r1	r1				

Shift –Reduce Conflict(衝突)

- Shift-Reduce Conflict はShift をすればいいのか，Reduceをすればいいのか曖昧な状況が存在することを表す.

(1) <E> → 1 <E>

(2) <E> → 1

アイテム集合

E → 1 • E

E → 1 •

+ E → • 1 E

+ E → • 1

この状態で1を読むと，shiftをすればいいか，還元をすればいいかが不明

Reduce –Reduce Conflict

- 2通りの還元の方法があり，どちらを行えばよいかがわからないケース.

(1) <E> → <A> 1

(2) <E> → <B> 2

(3) <A> → 1

(4) <B> → 1

アイテム集合 1

A → 1・

B → 1・

この状態で1を読むと，AとBのいずれに還元していいかが決まらない.

これらの問題の解決法

- Simple-LR (SLR)法　入力の次の記号が A の Follow Set に含まれる場合、規則 $A \rightarrow w$ を使い還元を行う.

- Look Ahead LR(LALR)法　入力の次の記号がAのLookahead Setに含まれる場合規則 $A \rightarrow w$ を適用する.

Follow-set と Lookahead-set

- 状態 S でのアイテム iの Follow-set（follow(i)）は、文法上 iの左辺の非終端記号の直後に出現可能な全記号から成る.
- 状態 S でのアイテム iの Lookahead-set は、状態 S で構文解析を開始したとき i の右辺に出現可能な記号のみを含む.
- *follow*(i) は左辺が同じ i である全 LR(0)アイテムの Lookahead-set の和集合と等価であり、状態やアイテムの右辺は考慮されていない.

これ以上は難しくなりすぎるので略

- 詳しく知りたい人は,

  コンパイラ（Principles of Compiler Design）, A.V.Aho, J.D.Ullman著／土居範久監訳（培風館）

  などを参考にして調べてみてください.
- 第3セメスタ開講科目,「データ構造とプログラミング技法」でも続きの説明を行います.

求めた表を使った構文解析例 1+1*0

状態	入力	スタック		意味
0	1+1*0$	[0]	s2	(1)+1*0
2	+1*0$	[2,0]	r6	<T>+1*0
3	+1*0$	[3,0]	r4	<B>+1*0
5	+1*0$	[5,0]	r2	<E>+1*0
4	+1*0$	[4,0]	s6	<E>(+)1*0
6	1*0$	[6,4,0]	s2	<E>(+)(1)*0
2	*0$	[2,6,4,0]	r6	<E>(+)<T>*0
3	*0$	[3,6,4,0]	r4	<E>(+)<B>*0
9	*0$	[9,6,4,0]	s7	<E>(+)<B>(*)0
7	0$	[7,9,6,4,0]	s1	<E>(+)<B>(*)(0)
1	$	[1,7,9,6,4,0]	r5	<E>(+)<B>(*)<T>
8	$	[8,7,9,6,4,0]	r3	<E>(+)<B>
9	$	[9,6,4,0]	r1	<E>
4	$	[4,0]	acc	終了

	アクション						GOTO		
状態	*	+	0	1	$		E	B	T
0			s1	s2			4	5	3
1	r5	r5	r5	r5	r5				
2	r6	r6	r6	r6	r6				
3	r4	r4	r4	r4	r4				
4		s6			acc				
5	s7 r2	r2	r2	r2	r2				
6			s1	s2				9	3
7			s1	s2					8
8	r3	r3	r3	r3	r3				
9	s7 r1	r1	r1	r1	r1				

導出木

(1) <E> →<E>+<B>

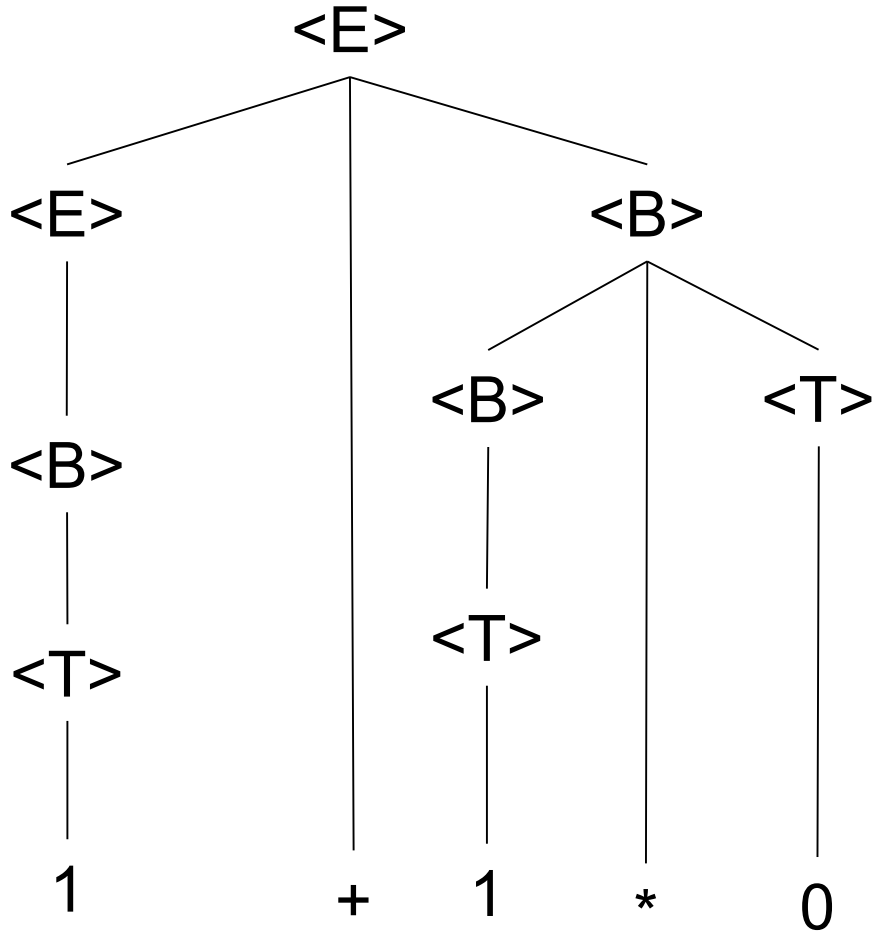
(2) <E> →<B>

(3) <B> →<B>*<T>

(4) <B> →<T>

(5) <T> → 0

(6) <T> → 1



まとめ

- 字句解析
  - オートマトンの機械的な作成法
- 構文解析
  - 最左導出を前提とした下降型構文解析（LL）
    - 構文解析表とスタックの利用
  - 最右導出を前提とした上昇型構文解析（LR）
    - アクション表とGOTO表とスタックの利用
    - Shift-reduce conflict, reduce-reduce conflict を回避する2つの方法SLR,LALR法