Compilers 第 3 次作業

2021.06.02

00857113 資工3B 蕭如珊

* 作業請以 pdf 格式上傳至 TronClass，或者以手寫方式採掃描或拍照上傳，但請注意圖片解析度。
  + 上傳作業之檔案請依下列規則命名：”學號-作業編號.副檔名”
    - For example: 00457000-hw3.pdf
  + 繳交期限為 6/10(五)下午 5 點前。
  + 答案將於 6/10(五)下午 5 點後開放。
* 答案需說明原因或計算過程。

1. (15%) Multiple Selection Questions:
   1. ( **4** ) A bottom-up parser generates: (1)Left-most derivation (2)Left-most derivation (3)Left-most derivation in reverse (4)Right-most derivation in reverse

**By the example and the sentence: “That is, bottom-up parsing during a left-to-right scan of the input constructs a right-most derivation in reverse.” of the textbook *Part3-Syntax Analysis(下)* page 4, we can know the answer is (4).**

* 1. ( **4** ) Consider the following statements related to compiler construction: I. Lexical Analysis is specified by context-free grammars and implemented by pushdown automata. II. Syntax Analysis is specified by regular expressions and implemented by finite-state machine. Which of the above statement(s) is/are correct? (1)Only I (2)Only II (3)Both I and II (4)Neither I nor II

**The correct statements should be:**

**“Lexical Analysis is specified by regular expressions and implemented by finite-state machine.” and “Syntax Analysis is specified by context-free grammars and implemented by pushdown automata.”, so the answer is (4).**

* 1. ( **3** ) Which of these is true about LR parsing? (1)Is most general non-backtracking shift- reduce parsing (2)It is still efficient (3)Both a and b (4)None of the mentioned

**By the advantages of LR parsing that has been mentioned in the textbook *Part3-Syntax Analysis(下)* page 13, we can know the answer is (3).**

* 1. ( **4** ) If a state does not know whether it will make a reduction operating using the production rule *i* or *j* for a terminal is called (1)Reduce conflict (2)Shift conflict (3)Shift/reduce conflict (4)Reduce/reduce conflict

**During shift-reduce parsing, there are two kinds of conflicts that often occur: the “shift/reduce conflict” and the “reduce/reduce conflict”. The shift/reduce conflict means that we cannot decide to shift or to reduce and the reduce/reduce conflict means that there are several reductions we can take so we can’t decide which should be taken.**

* 1. ( **1**) The construction of the canonical collection of the sets of LR(1) items are similar to the construction of the canonical collection of the sets of LR(0) items. Which is an exception? (1)Closure and goto operations work a little bit different (2)Closure and goto operations work similarly (3)Closure and additive operations work a little bit different (4)Closure and associatively operations work a little bit different

**To talk about the differences of the construction of the canonical collection of the sets of LR(0) and LR(1). From the general form of the items, there begin to have a little bit difference in LR(0) and LR(1). In LR(0), the general form of item is [A→α‧Bβ] and the general form of item in LR(1) is [A→α‧Bβ, a] which do an extra action called “lookahead”. So in LR(1), the CLOSURE() method have to do an additional for-loop to find FIRST(βa) and add into the tail of the item that separated by a comma.**

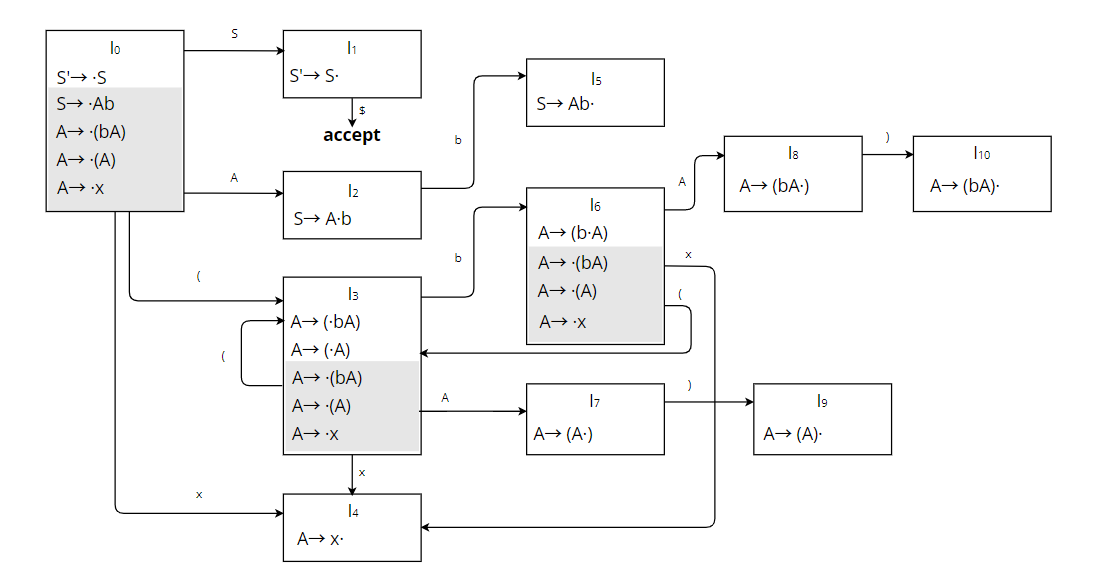
1. (50%) For the following grammar, S → Ab

A → (bA)

A → (A)

A → x

* 1. (15%) construct the LR(0) automaton of this grammar



* 1. (10%) construct the LR(0) parsing table of this grammar

*S’→S ,* (1)*S → Ab* (2)*A → (bA)* (3)*A → (A)*  (4)*A → x*

|  |  |  |
| --- | --- | --- |
| **STATE** | **ACTION** | **GOTO** |
| **b ( ) x $** | **S A** |
| **0** | s3 s4 | 1 2 |
| **1** | acc |  |
| **2** | s5 |  |
| **3** | s6 s3 s4 | 7 |
| **4** | r4 r4 r4 r4 r4 |  |
| **5** | r1 r1 r1 r1 r1 |  |
| **6** | s3 s4 | 8 |
| **7** | s9 |  |
| **8** | s10 |  |
| **9** | r3 r3 r3 r3 r3 |  |
| **10** | r2 r2 r2 r2 r2 |  |

* 1. (10%) construct the SLR(1) parsing table of this grammar

FOLLOW(S’) = {$}

FOLLOW(S) = {$}

FOLLOW(A) = {b, )}

|  |  |  |
| --- | --- | --- |
| **STATE** | **ACTION** | **GOTO** |
| **b ( ) x $** | **S A** |
| **0** | s3 s4 | 1 2 |
| **1** | acc |  |
| **2** | s5 |  |
| **3** | s6 s3 s4 | 7 |
| **4** | r4 r4 |  |
| **5** | r1 |  |
| **6** | s3 s4 | 8 |
| **7** | s9 |  |
| **8** | s10 |  |
| **9** | r3 r3 |  |
| **10** | r2 r2 |  |

* 1. (15%) parse the string (b(x))b$ by using the SLR(1) parsing table

0 ( 3 b 6 ( 3 x 4 ) → r4: reduce x by A

0 ( 3 b 6 ( 3 A 7 ) 9 ) → r3: reduce (A) by A

0 ( 3 b 6 A 8 ) 10 b → r2: reduce (bA) by A

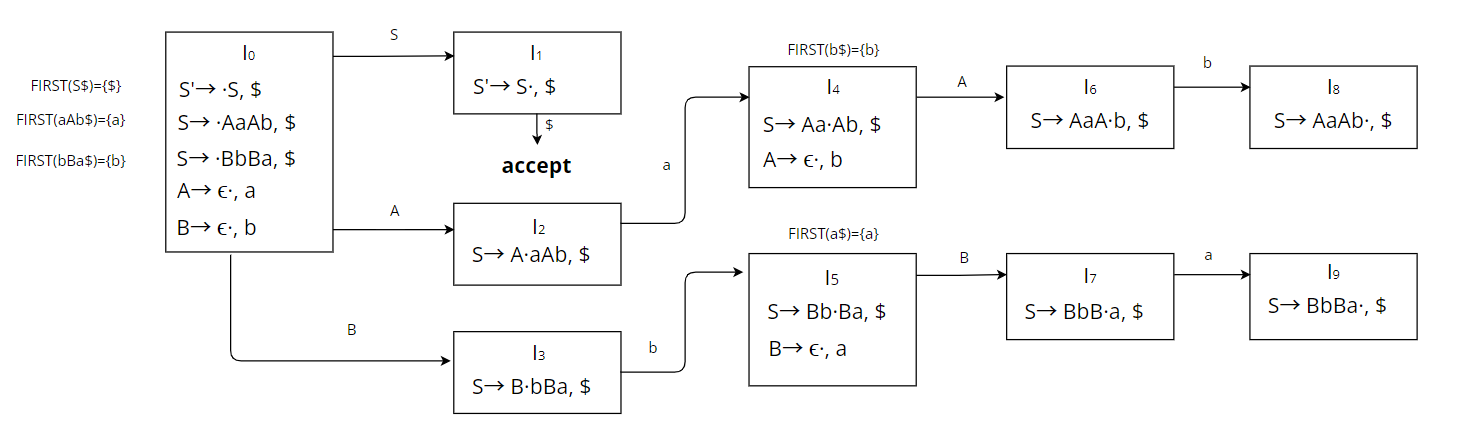
0 A 2 b 5 $ → r1: reduce Ab by S

0 S 1 $ => accept

1. (35%) For the following grammar, S  A **a** A **b** | B **b** B **a**

A  ϵ B  ϵ

* 1. (15%) Construct the LR(1) automaton of the grammar



* 1. (10%) Construct the LR(1) parsing table of the grammar

S→S’, (1)*S → Aa A b | B b B a* (2)*A → ϵ* (3)*B → ϵ*

|  |  |  |
| --- | --- | --- |
| **STATE** | **ACTION** | **GOTO** |
| **a b $** | **S A B** |
| **0** | r2 r3 | 1 2 3 |
| **1** | acc |  |
| **2** | s4 |  |
| **3** | s5 |  |
| **4** | r2 | 6 |
| **5** | r3 | 7 |
| **6** | s8 |  |
| **7** | s9 |  |
| **8** | r1 |  |
| **9** | r1 |  |

* 1. (10%) Construct the LALR(1) parsing table of the grammar

S→S’, (1)*S → Aa A b | B b B a* (2)*A → ϵ* (3)*B → ϵ*

沒有可合併的狀態。

|  |  |  |
| --- | --- | --- |
| **STATE** | **ACTION** | **GOTO** |
| **a b $** | **S A B** |
| **0** | r2 r3 | 1 2 3 |
| **1** | acc |  |
| **2** | s4 |  |
| **3** | s5 |  |
| **4** | r2 | 6 |
| **5** | r3 | 7 |
| **6** | s8 |  |
| **7** | s9 |  |
| **8** | r1 |  |
| **9** | r1 |  |