

编译原理(A)大作业
算符优先分析表和卷积优化

李子龙 518070910095

2021 年 6 月 15 日

目录

| | |
|----------------------|---|
| 第一部分 算符优先分析表 | 2 |
| 1 题目分析 | 3 |
| 2 算法描述 | 4 |
| 2.1 集合依赖关系 | 4 |
| 2.2 三进 DFS | 4 |
| 第二部分 卷积优化 | 4 |

第一部分 算符优先分析表

运行环境

操作系统 Windows

语言 Rust[1]

```
opg [filename]
```

程序输出

样例 1 输入

```
E -> E + T | T
T -> T * F | F
F -> ( E ) | i
```

样例 1 输出

| | * | (|) | + | i | \$ |
|----|---|---|---|---|---|----|
| * | > | < | > | > | < | > |
| (| < | < | = | < | < | |
|) | > | | > | > | | > |
| + | < | < | > | > | < | > |
| i | > | | > | > | | > |
| \$ | < | < | | < | < | = |

样例 2 输入

```
E -> E + E | E * E | ( E ) | id
```

样例 2 输出

The grammar is ambiguous.

1 题目分析

首先需要声明在算符优先语法中算符优先级的定义[2]。

定义 1 对于两个终结符 T_1 和 T_2 ，有下面的算符优先级定义（其中 U_1 是非终结符）

1. $T_1 = T_2$ 如果存在产生式 $U \rightarrow xT_1T_2y$ 或 $U \rightarrow xT_1U_1T_2y$ 。
2. $T_1 < T_2$ 如果存在产生式 $U \rightarrow xT_1U_1y$ 而且存在一个推导 $U_1 \Rightarrow z$ 使得 T_2 是 z 的最左终结符。
3. $T_1 > T_2$ 如果存在产生式 $U \rightarrow xU_1T_2y$ 而且存在一个推导 $U_1 \Rightarrow z$ 使得 T_1 是 z 的最右终结符。

本部分即针对一个上下文无关文法，输出算符优先分析表。如果文法是有二义性的，将会报错。如果两个终结符之间没有上述三个关系的其中一个，将会留空，意为没有优先关系。

根据定义 1，可以对符号构造下面两个集合以判断情况 2 和 3：

定义 2 假设 V_T 是该文法终结符号对应的集合， V_N 是该文法非终结符号对应的集合。对符号 U_1 定义下面两个集合：

$$FIRSTVT(U_1) = \{T | (U_1 \Rightarrow Ty \vee U_1 \Rightarrow U_2Ty) \wedge T \in V_T \wedge U_2 \in V_N\} \quad (1)$$

$$LASTVT(U_1) = \{T | (U_1 \Rightarrow xT \vee U_1 \Rightarrow xTU_2) \wedge T \in V_T \wedge U_2 \in V_N\} \quad (2)$$

除了上面的定义方法，对于这样的集合，还有这样的性质：

$$(U_1 \Rightarrow U_2y) \Rightarrow FIRSTVT(U_2) \subseteq FIRSTVT(U_1) \quad (3)$$

$$(U_1 \Rightarrow xU_2) \Rightarrow LASTVT(U_2) \subseteq LASTVT(U_1) \quad (4)$$

这样定义 1 就有了如下的等价定义：

定义 3 对于两个终结符 T_1 和 T_2 ，有下面的算符优先级定义（其中 U_1 是非终结符）

1. 如果找到了这样的产生式右部 T_1T_2 或 $T_1U_1T_2$ ，那么 $T_1 = T_2$ 。
2. 如果找到了这样的产生式右部 T_1U_1 且 $T_2 \in FIRSTVT(U_1)$ ，那么 $T_1 < T_2$ 。
3. 如果找到了这样的产生式右部 U_1T_2 且 $T_1 \in LASTVT(U_1)$ ，那么 $T_1 > T_2$ 。

2 算法描述

2.1 集合依赖关系

首要任务就是对于每一个非终结符求出 $FIRSTVT$ 和 $LASTVT$ 。式 (1) 和 (2) 所对应的终结符为图中的盲端，式 (3) 和 (4) 所对应的非终结符导出的依赖节点为中间节点。这样就可以构造出集合的依赖关系有向图。

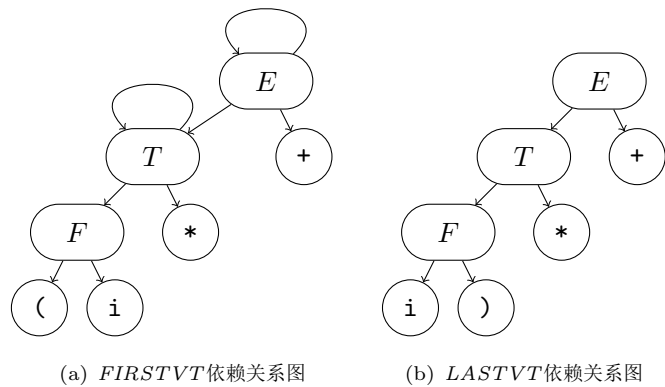


图 1: 依赖关系图

2.2 三进 DFS

一进 DFS:

第二部分 卷积优化

优化结果

| 输入大小和输出大小 | 优化前 | 优化后 | 提升效率 |
|------------------------------------|-----|-----|------|
| $n, ic, ih, iw = 1, 3, 32, 32$ | | | |
| $oc, kh, kw = 32, 3, 3$ | | | |
| $n, ic, ih, iw = 100, 512, 32, 32$ | | | |
| $oc, kh, kw = 1024, 3, 3$ | | | |

参考文献

- [1] C. N. Steve Klabnik, *The Rust Programming Language (Covers Rust 2018)*. Random House LCC US, 2019. [Online]. Available: https://www.ebook.de/de/product/37149179/steve_klabnik_carol_nichols_the_rust_programming_language_covers_rust_2018.html
- [2] R. W. Floyd, “Syntactic analysis and operator precedence,” *Journal of the ACM*, vol. 10, no. 3, pp. 316–333, Jul. 1963.