



编译原理课设报告

专 业： 计算机科学与技术

班 级： 14-3 班

学 号： 2014211674

姓 名： 欧阳杰

（一）课程设计任务、要求、目的

任务：

消除回溯算法的程序实现

设计内容及要求：

构造一程序，实现：消除文法每一条产生式候选式的公共左因子。对于用户任意输入的文法 G ，输出一个无回溯的等价文法，可显示输出，或输出到指定文件中。

目的：

对于用户任意输入的文法 G ，输出一个无回溯的等价文法，可显示输出，或输出到指定文件中。

（二）原理及算法描述

一个文法由多个产生式组成，依次对文法中的每个产生式进行消除公共左因子。我消除公共左因子的做法是：首先将首字母相同的候选式放到一个集合中，然后再依次检查这些同个集合中的候选式的第二个字母是否相同，循环下去，直至不同。这样就可以得到最长公共左因子，然后引入新的推导式进行公共左因子的消除。

（三）开发环境

OS : WIN 10

编辑器 : Pycharm

语言 : python

(四) 重要算法和设计思路描述

1. 对前端传过来的文法进行格式化, 得到我们需要的格式, python 里的字典, 形如

$G = \{ 'A': ['abc', 'ab', 'abcd', 'a', 'b', 'bc'] \}$

```
def format(text):  
    d = dict()  
    r = text.split('\n')  
    for t in r:  
        if t is not "":  
            l = t.split()  
            d[l[0]] = l[1:]  
    return d
```

2. 对消除公共左因子后的文法进行格式化, 然后传给前端展示, 形如:

$A \rightarrow cA' | abcA''$

$A' \rightarrow \epsilon | d$

$A'' \rightarrow \epsilon | d$

```
def output(d):  
    s = ""  
    for k, v in d.items():  
        str = k + '->' + '|'.join(v)  
        s += str + '\n'  
    return s
```

3. 消除公共左因子算法

对每一个产生式,将具有公共左因子的候选式提取出来公共左因子引

入新的产生式,同时原产生式将这些具有公共左因子的候选式删除,

引入“公共左因子+引入的非终结符”的候选式

```
def eliminate_common_factor(VN, exp_list):
    # G:  {'A': ['abc', 'ab', 'abcd', 'a', 'b', 'bc', 'B']}
    index_dict = get_common_factor(exp_list)
    if have_common_factor(index_dict):
        for k, v in index_dict.items():
            len_k = len(k)
            tmp_list = []
            if len(v) != 1:
                for i in v:
                    exp = exp_list[i][len_k:]
                    tmp_list.append(exp if exp else 'ε')
            new_VN = generate_VN(VN)
            exp_list.append(k + new_VN) # 加入 aA'
            G_c[new_VN] = tmp_list # 将新生成的规则添加进 G_c 文法中
    remove_exp(index_dict, exp_list)
    # print('VN:', VN)
    result[VN] = exp_list
    # print('result:', result)
    G_c.pop(VN)
```

(五) 程序实现---程序清单

后端：

```
# -*- coding: utf-8 -*-
```

```
"""
```

```
@author: W@I@S@E
```

```
@contact: wisecsj@gmail.com
```

```
@site: http://hfutoyj.cn/
```

```
@file: app.py
```

```
@time: 2017/9/8 9:01
```

```
"""
```

```
from flask import Flask, render_template, jsonify, request
from utils import wise
```

```
app = Flask(__name__)
```

```
def format(text):
    d = dict()
    r = text.split('\n')
    for t in r:
        if t is not "":
            l = t.split()
            d[l[0]] = l[1:]
    return d
```

```
def output(d):
    s = ""
    for k, v in d.items():
        str = k + '->' + '|'.join(v)
        s += str + '\n'
    return s
```

```
@app.route('/')
def index():
    return render_template('index.html')
```

```
@app.route('/eliminate', methods=['POST'])
def eliminate():
    G = request.json.get('G', None)
    d = format(G)
    print(d)
    result = wise(d)
    finally_result = output(result)
    return jsonify(G_r=finally_result)
```

```
if __name__ == '__main__':
    app.run(host='127.0.0.1', port=5000, debug=True)
```

消除公共左因子算法代码：

```
import copy
from collections import defaultdict

# G = {'A': ['abc', 'ab', 'abcd', 'a', 'b', 'bc', 'B']}
# G['B'] = ['Ab', 'Ac']
# print(G)
VN_dict = defaultdict(int)
G_c = None
result = dict()

def eliminate_common_factor(VN, exp_list):
    # G: {'A': ['abc', 'ab', 'abcd', 'a', 'b', 'bc', 'B']}
    index_dict = get_common_factor(exp_list)
    if have_common_factor(index_dict):
        for k, v in index_dict.items():
            len_k = len(k)
            tmp_list = []
            if len(v) != 1:
                for i in v:
                    exp = exp_list[i][len_k:]
                    tmp_list.append(exp if exp else 'ε')
                new_VN = generate_VN(VN)
                exp_list.append(k + new_VN) # 加入 aA'
                G_c[new_VN] = tmp_list # 将新生成的规则添加进 G_c 文法中
    remove_exp(index_dict, exp_list)
    # print('VN:', VN)
    result[VN] = exp_list
    # print('result:', result)
    G_c.pop(VN)

def remove_exp(index_dict, exp_list):
    l = []
    for v in index_dict.values():
        if len(v) != 1:
            l.extend(v)
    # 在对一个 list 进行 pop 操作时，必须是按逆序来删除。
    # 即，先删下标大的元素，再删下标小的元素，否则会越界。
    l = sorted(l, reverse=True)
```

```

        for index in l:
            exp_list.pop(index)

def get_common_factor_aux(exp_list):
    """
    对于 G = {'A': 'abc ab abcd a b bc'}
    :param exp_list: 文法 G 中每条产生式右部的集和的列表形式: ['abc', 'ab', 'abcd',
    'a', 'b', 'bc']
    :return: 类型为字典, key 为首字符, value 为所在 index: {'b': [4, 5], 'a': [0, 1, 2,
    3]})
    """
    d = defaultdict(list)
    for index, exp in enumerate(exp_list):
        d[exp[0]].append(index)
    return d

def get_common_factor(exp_list):
    d = get_common_factor_aux(exp_list)
    index = 1
    d_copy = copy.deepcopy(d)
    for k, v in d.items():
        new_k = copy.deepcopy(k)
        k_copy = copy.deepcopy(k)
        while True:
            try:
                c = exp_list[v[0]][index]
            except Exception as e:
                break
            for i in v:
                if exp_list[i][index] == c:
                    pass
                else:
                    break
            else:
                new_k = k_copy + c
                d_copy[new_k] = v
                del d_copy[k_copy]
                index += 1
                k_copy = copy.deepcopy(new_k)
                continue
            if new_k == k_copy:
                break

```

```
return d_copy
```

```
def have_common_factor(d):  
    for v in d.values():  
        if len(v) != 1:  
            return True  
    else:  
        return False
```

```
def generate_VN(VN):  
    VN_dict[VN[0]] += 1  
    return VN[0] + VN_dict[VN[0]] * ""
```

```
def wise(G):  
    global G_c, VN_dict, result  
    VN_dict = defaultdict(int)  
    G_c = copy.deepcopy(G)  
    result = dict()  
    while True:  
        size = len(result)  
        try:  
            for k, v in G.items():  
                eliminate_common_factor(k, v)  
        except Exception as e:  
            raise e  
        G = copy.deepcopy(G_c)  
        if size == len(result):  
            break  
    return result
```

```
if __name__ == '__main__':  
    G = {'A': ['abc', 'ab', 'abcd', 'a', 'b', 'bc', 'B']}  
    G_c = copy.deepcopy(G)  
    while True:  
        size = len(result)  
        try:  
            for k, v in G.items():  
                eliminate_common_factor(k, v)  
        except Exception as e:  
            raise e
```



```

G = copy.deepcopy(G_c)
if size == len(result):
    break
print(result)

```

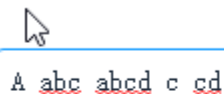
(六) 总结

通过这次课设，我了解如何通过不动点算法来循环消除公共左因子以确
保文法是无需回溯的。同时，这次我使用了 B/S 架构。通过再前端界面
输入文法，后端处理进行消除得到结果并返回前端，前端再来进行展示
的方式。熟悉了前后端的运作方式。

(七) 测试结果

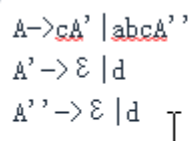
测试 1：

文法：



A abc abcd c cd

结果：



$A \rightarrow cA' \mid abcA''$
 $A' \rightarrow \varepsilon \mid d$
 $A'' \rightarrow \varepsilon \mid d$

测试二：

文法：



A abc ab abcd a b bc B

结果：

W

$A' \rightarrow \varepsilon \mid c$

$A' \rightarrow \varepsilon \mid bA''$

$A''' \rightarrow \varepsilon \mid d$

$A \rightarrow B \mid aA' \mid bA''$

$A''' \rightarrow \varepsilon \mid cA'''$