词法分析设计文档

18373054 陆晓东

2020 年 9 月 24 日

1 编码前设计

1.1 识别单词划分

对于需要识别的单词，可以根据单词的特性进行划分。这里主要按照树状结构进行了划分。然后对于不同划分层次的单词进行不同的识操作和处理操作。

首先可以将只需读入一个字符就能判定类别的单字符分为一类，叫做单符号判定类。其他类别叫做多符号判断类。在多符号判断类中，first集合独一无二且长度固定的类别叫做直接可判断类(!=等)。first有重复且长度固定的称为前缀相同类。长度不固定的为循环读入类。循环读入类有字符常量，字符串，标识符&保留字类。

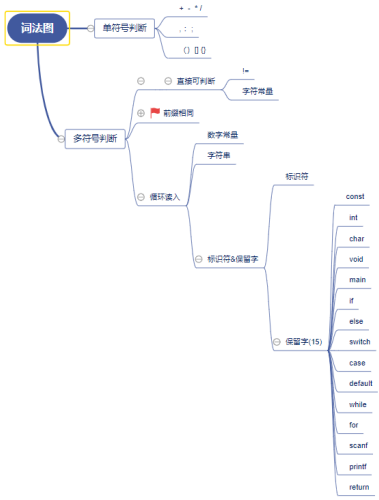

图 1: 分类示意图

1.2 编码方案

根据不同分类采取不同的处理方案

1. 单字符可以分隔:直接读入即可
2. 直接可判定类:

(a) != : 再读入一个字符即可，若不是=则直进入错误处理，否则转出

(b) 字符常量:读入两个字符c1,c2,若c2不为'或者c1不为字符常量，则进入错误处理，否则转出

3. 前缀相同：再读入一个字符,若为=则当作前缀相同中长的处理，否则当作短的处理并且退回一个单词

4. 循环读入类

(a) 数字常量：重复读入知道不为0 – 9 之间的单词，最后吐出一个

(b) 字符串:重复读入直到不合法字符串组成元素，若最后读到字符不为"则进入错误处理，否则退出

(c) 标识符&保留字:重复读入直到不为数字或者字符，退回最后读入的一个。和保留字一一比较，若有相同的则当保留字处理，否则当标识符处理

2 编码完成后修改

编码完成后主要对读入eof后退回字符可能导致多次读入的情况，增加了对eof的处理。(主要针对不符合文法情况的代码)

Listing 1: 修改后核心代码

```
 int c=0;
string ms;
while(true){
ms.clear();
c = fgetc(fp);
while(c==' '||c=='\n'||c=='\t'||c=='\r'){
c = fgetc(fp);
}
if(c==-1) break;
//@signle
if(c=='('){
ms.push_back(c);
add(LPARENT,ms);
}else if(c==')'){
ms.push_back(c);
add(RPARENT,ms);
}else if(c=='['){
ms.push_back('[');
add(LBRACK,ms);
}else if(c==']'){
ms.push_back(']');
add(RBRACK,ms);
}
else if(c=='{'){
ms.push_back(c);
add(LBRACE,ms);
}else if(c=='}'){
ms.push_back(c);
```

```
29          add(RBRACE,ms);
30          }else if(c=='+'){
31          ms.push_back(c);
32          add(PLUS,ms);
33          }else if(c=='-'){
34          ms.push_back(c);
35          add(MINU,ms);
36          }else if(c=='*'){
37          ms.push_back(c);
38          add(MULT,ms);
39          }else if(c=='/'){
40          ms.push_back(c);
41          add(DIV,ms);
42          }else if(c==','){
43          ms.push_back(c);
44          add(COMMA,ms);
45          }else if(c==':'){
46          ms.push_back(c);
47          add(COLON,ms);
48          }else if(c==';'){
49          ms.push_back(c);
50          add(SEMICN,ms);
51          }
52          //@doubel+
53          else if(c=='!'){
54          c = fgetc(fp);
55          if(c!='=') error(0);
56          else{
57          ms.append("!=");
58          add(NEQ,ms);
59          }
60          }
61          else if(c=='='){ // ==\=
```

```
62          ms.push_back(c);
63          c = fgetc(fp);
64          if(c=='='){
65          ms.push_back(c);
66          add(EQL,ms);
67          }else{
68          fseek(fp,-1,SEEK_CUR);
69          add(ASSIGN,ms);
70          }
71          }
72          else if(c=='<'){ //<\<=
73          ms.push_back(c);
74          c = fgetc(fp);
75          if(c=='='){
76          ms.push_back(c);
77          add(LEQ,ms);
78          }else{
79          fseek(fp,-1,SEEK_CUR);
80          add(LSS,ms);
81          }
82          }
83          else if(c=='>'){ // >\>=
84          ms.push_back(c);
85          c = fgetc(fp);
86          if(c=='='){
87          ms.push_back(c);
88          add(GEQ,ms);
89          }else{
90          fseek(fp,-1,SEEK_CUR);
91          add(GRE,ms);
92          }
93          }
94          else if(c=='\"'){
```

```
95          c = fgetc(fp);
96          while(ISSTR(c)){
97          ms.push_back(c);
98          c = fgetc(fp);
99          }
100
101         if(c!='\"') error(0);
102         else add(STRCON,ms);
103
104         }else if(c=='\''){
105         char mc1=fgetc(fp),mc2=fgetc(fp);
106         if(mc2!='\''||!ISCHAR(mc1)){
107         error(0);
108         }else{
109         ms.push_back(mc1);
110         add(CHARCON,ms);
111         }
112         }
113         //@multi
114         else if(ISDIGIT(c)){
115         ms.push_back(c);
116         c= fgetc(fp);
117         while(ISDIGIT(c)){
118         ms.push_back(c);
119         c = fgetc(fp);
120         }
121         fseek(fp,-1,SEEK_CUR);
122         add(INTCON,ms);
123         }
124         else if(ISALPHA(c)){ //15 special && 1 common
125         ms.push_back(c);
126         c = fgetc(fp);
127         while(ISALPHA(c)||ISDIGIT(c)){
```

```
128         ms.push_back(c);
129         c= fgetc(fp);
130         }
131         string mms = ms;
132         for(auto it=mms.begin();it!=mms.end();it++){
133         *it = LOWER((*it));
134         }
135         if(mms.compare("const")==0){
136         add(CONSTTK,ms);
137         }else if(mms.compare("int")==0){
138         add(INTTK,ms);
139         }else if(mms.compare("char")==0){
140         add(CHARTK,ms);
141         }else if(mms.compare("void")==0){
142         add(VOIDTK,ms);
143         }else if(mms.compare("main")==0){
144         add(MAINTK,ms);
145         }else if(mms.compare("if")==0){
146         add(IFTK,ms);
147         }else if(mms.compare("else")==0){
148         add(ELSETK,ms);
149         }else if(mms.compare("switch")==0){
150         add(SWITCHTK,ms);
151         }else if(mms.compare("case")==0){
152         add(CASETK,ms);
153         }else if(mms.compare("default")==0){
154         add(DEFAULTTK,ms);
155         }else if(mms.compare("while")==0){
156         add(WHILETK,ms);
157         }else if(mms.compare("for")==0){
158         add(FORTK,ms);
159         }else if(mms.compare("scanf")==0){
160         add(SCANFTK,ms);
```

```
161         }else if(mms.compare("printf")==0){
162         add(PRINTFTK,ms);
163         }else if(mms.compare("return")==0){
164         add(RETURNTK,ms);
165         }
166         else{
167         add(IDENFR,ms);
168         }
169         if(c!=-1) fseek(fp,-1,SEEK_CUR); //eof pd
170         }else{
171         error(0);
172         }
173
174         }
```