Dictionnaire

Philippe Langevin

Novembre 2007

Abstract

Le document est basé sur le programme de Valentin Schmitt. Il illustre l'insertion de mots dans un dictionnaire. La commande `dot` est utilisée pour obtenir une représentation graphique. Le tout est trés sommaire, idéalement l'étudiant devrait compléter la source `dico.tex` pour obtenir un document correctement finalisé.

Contents

1 analyse.lex

1.1 Déclarations C:construction d'un arbre

```
%{
#include <stdio.h>
#include <stdlib.h>
  int mot=0, numero=0;
typedef unsigned long ulong;
typedef struct tnoeud_ {
    char c;
    ulong n;
    int   num;
    struct tnoeud_ *fa,*fd;
    } tnoeud, * tarbre;
tarbre arbre = NULL;
char buffer[256]={0};

tarbre new(char c, ulong n, tarbre fa, tarbre fd) {
    tarbre aux;
    aux = (tarbre) malloc (sizeof(tnoeud));
    aux->c = c;
    aux->n = n;
    aux->fa = fa;
    aux->fd = fd;
    aux->num = numero++;
    return aux;
    }

tarbre inserer(tarbre * arbre, char * mot) {
    tarbre aux;
    if(*arbre==NULL) *arbre=new(*mot, 0 ,NULL,NULL);
    aux=*arbre;
    while( 1 ) {
        if( aux->c==*mot) {
            mot++;
            if( *mot== 0) {
                aux->n++;
                return aux;
               }
            if( ! aux->fa) aux->fa=new(*mot, 0 ,NULL,NULL);
            aux = aux->fa;
            }
        else {
            if( ! aux->fd) aux->fd=new(*mot, 0 ,NULL,NULL);
            aux = aux -> fd;
            }
        }
```

```
    }
```

1.2 Déclarations C:affichages d'un arbre

```
void affiche(tarbre arbre, int pos) {
    while(arbre) {
        buffer[pos]=arbre->c;
        if(arbre->n) printf("\n%s (%ld)",buffer, arbre->n);
        affiche(arbre->fa, pos+1);
        arbre=arbre->fd;
        buffer[pos]=0;
        }
    }

 void dot( tarbre arbre , int mode)
 { tarbre aux;
  if ( arbre == NULL ) return;
   printf("\n\"node%d\" [ color=red , label = \"<f0>%c|<f1>%lu\", shape = \"record\" ];
   if ( arbre->fa )
     printf("\n\"node%d\":f0 -> \"node%d\":f0 [color=green];", arbre->num, arbre->fa->n
   aux = arbre;
   if ( mode )
     while ( aux->fd ) {
       printf("\n\"node%d\":f1 -> \"node%d\":f0 [ color=blue];", aux->num, aux->fd->num
       dot (  aux->fd , 0);
       aux = aux->fd;
   }
   dot (  arbre->fa, 1);
   }


void output( tarbre arbre )
{
    printf("\ndigraph {");
    printf("\n graph [ \n  ];");

    printf("\n node [ fontsize = \"16\" shape = \"ellipse\" ];");
    printf("\n size=\"18,10\"");
    dot( arbre , 1);
    printf("\n}");
}
```

1.3 Déclarations lex

```
%}

LETTRE   [A-Za-z]
MOT      {LETTRE}+
```

1.4 Règles

```
%%
{MOT}    inserer( &arbre, yytext) ;
.        ;
\n       ;
```

1.5 Appel de yylex

```
%%
int main(int argc, char* argv[] ) {
  int opt;
  int DOT = 0, TXT = 0;
  char * optliste = "s:dth";
     if ( argc > 1)
        yyin = fopen( argv[1], "r" );
    yyin = stdin;

  while (( opt = getopt( argc, argv, optliste )) >= 0 ) {
      switch ( opt ) {
        case 'd' : DOT = 1;     break;
        case 't' : TXT = 1;     break;
        case 's' : yyin = fopen( optarg , "r" );  break;
        case 'h' : printf("\nusage : -s src -dot -txt -help");
       default   : exit(1);
      }
  }
  yylex();
  if ( TXT ) affiche( arbre , 0);
  if ( DOT )  output( arbre );
  return(0);
}
```

2 Makefile

```
LIB=-lfl
CC=gcc
LL=flex
OPT=-Wall -g

dico.pdf : urls.tex sdic.pdf bdic.pdf analyse.lex dico.tex
        pdflatex dico.tex

analyse: analyse.lex
        $(LL) -oanalyse.c analyse.lex
        $(CC) $(OPT) -o analyse analyse.c $(LIB)

sdic.pdf : analyse essai.txt
        ./analyse -s essai.txt  -d > sdic.dot
```

```
        dot -Tfig sdic.dot -o sdic.fig
        fig2dev -Lpdf sdic.fig > sdic.pdf

bdic.pdf : analyse makefile
        ./analyse -s makefile -d > bdic.dot
        dot -Tfig bdic.dot -o bdic.fig
        fig2dev -Lpdf bdic.fig > bdic.pdf
urls    :
        ./urls.sh > urls.tex
```

3 Graphique

Les représentations sagittales des dictionnaires des mots contenus dans les fichiers `com.txt` (1) et `makefile` (2) sont construites par la commande `dot`.

4 Liens vers les sources

- makefile.sh urls.sh
- essai.txt
- analyse.lex
- analyse.c preprojet.c
- dico.tex urls.tex
- bigdic.fig big.fig smalldic.fig small.fig
- bigdic.pdf big.pdf dico.pdf smalldic.pdf small.pdf

References

[1] Jean-Pierre Zanotti Travaux-Pratiques d'Algorithmique Module I51, licence. http://zanotti.univ-tln.fr/enseignement/I51/TP5.html

[2] Le package "listings" ftp://ftp.inria.fr/pub/TeX/CTAN/macros/latex/contrib/listings/

[3] Le site "graphviz" http://www.graphviz.org/

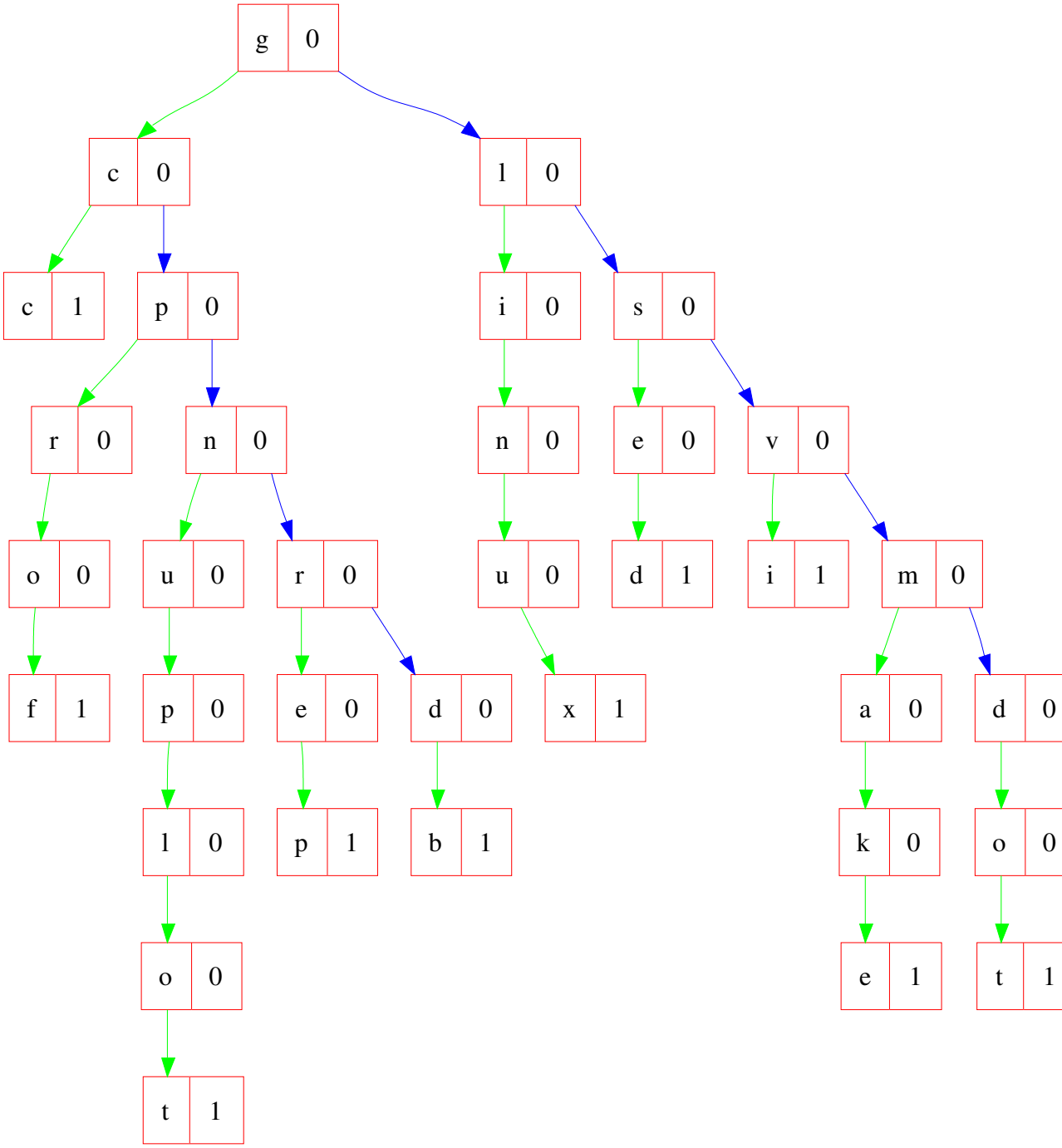

Figure 1: Exemple de dictionnaire

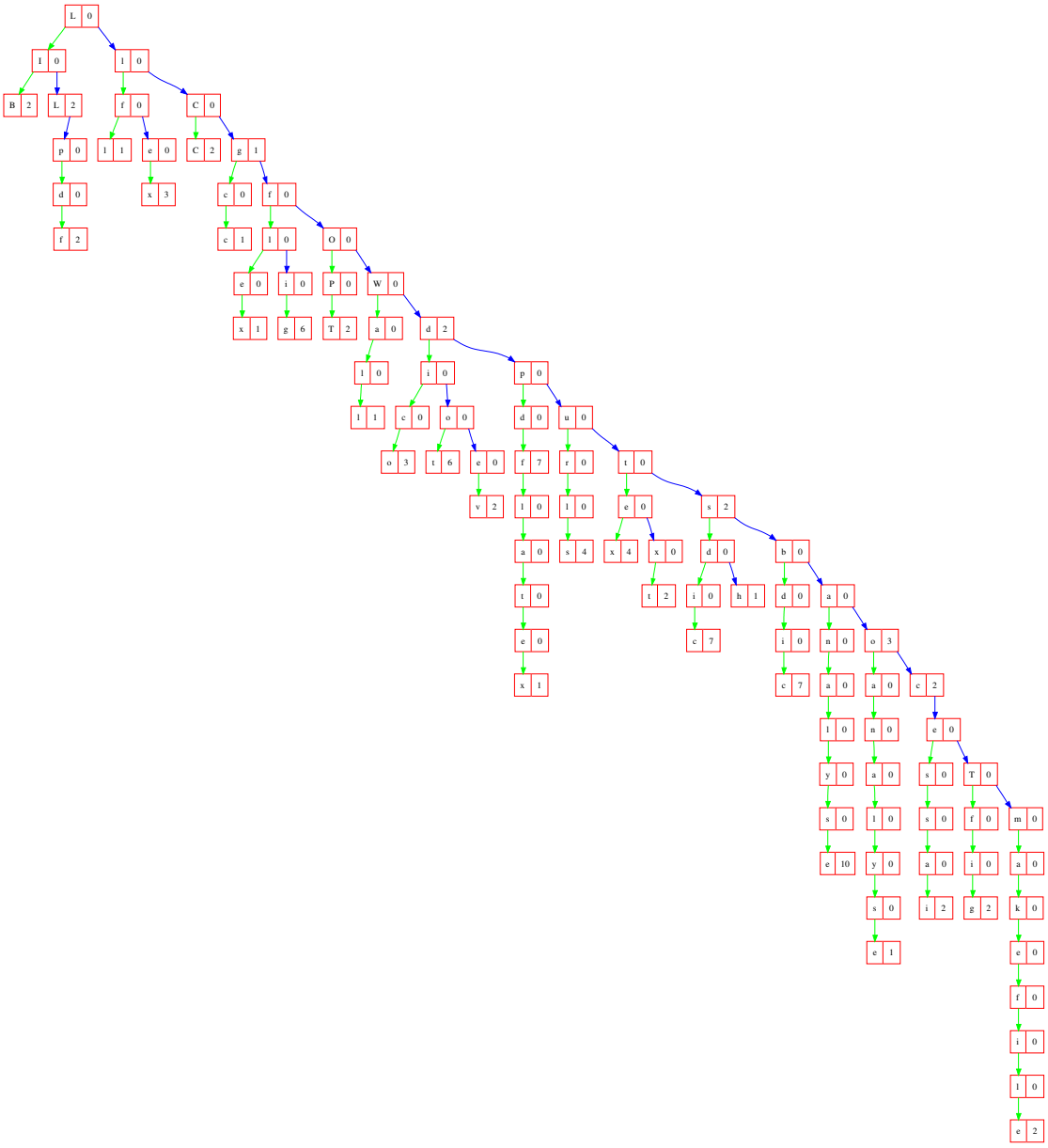

Figure 2: Exemple de dictionnaire