Incrémentation d'un entier

Philippe Langevin

Octobre 2007

Abstract

Le document est basé sur le programme de . Il illustre l'utilisation des outils de l'expérimentation numérique en langage C sous linux : `gcc`, `makefile`, `gnuplot`, `fig2dev`, et les packages listings et graphicx de `latex`. Le tout est trés sommaire, idéalement l'étudiant devrait compléter la source `inc.tex` pour obtenir un document correctement finalisé.

Contents

1 Programme

1.1 main

```
int main (int argc ,char*argv[])
  {
  int base=2, pas=1, nb, max=5;
  int opt;
  char * optliste = "p:b:m:h:v";
  while (( opt = getopt( argc, argv, optliste )) >= 0 ) {
      switch ( opt ) {
       case 'p' : pas  = atoi(optarg);   break;
       case 'b' : base = atoi(optarg);   break;
       case 'm' : max  = atoi(optarg);   break;
       case 'v' : verbose = 1;           break;
       case 'h' : printf("\nusage  %s: -p pas -m max -b base -verbose", argv[0]);
         printf("\ndefaut   : pas=1  max=5 base=2 verbose=0");
       default  : exit(1);
      }
  }
  output( base, pas, max);
  printf("\neoj...\n");
  return 0;
  }
```

La fonction `getopt` permet une gestion des options. L'expérience est réalisée par la fonction `output`.

1.2 output

```
void output(  int base , int pas , int max)
{ int nb, cpt, total, qte;
  nombre N;
  printf("\nbase=%d max=%d pas=%d", base, max, pas);
  for ( nb = pas; nb <= max; nb+= pas ) {
        N = zero ( nb );
        total = 0;
        qte = 0;
        do {
           cpt = plusun (N, base, nb);
           total += cpt;
           qte ++;

        } while ( cpt < nb );
        printf("\ntaille=%3d total=%7d moy=%.2f", nb, total, (float) total / qte);
  }
}
```

Une exécution est illustré par la figure 1.

```
base=3 max=10 pas=2
taille=  2 total=      4 moy=0.44
taille=  4 total=     40 moy=0.49
taille=  6 total=    364 moy=0.50
taille=  8 total=   3280 moy=0.50
taille= 10 total=  29524 moy=0.50
eoj...
```

Figure 1: Un exemple d'exécution: `./inc.exe -p 2 -b 3 -m 10`

1.3 procédures et fonctions

```
typedef int* nombre;
typedef int chiffre;
int verbose = 0;
void affiche (nombre z,int max)
  { int i;
    for(i= max - 1 ;i >= 0 ;i--)
       printf(" %d", z[i] );
   }

nombre zero (int max)
  {
  return (nombre) calloc (max, sizeof( chiffre ));
  }

int plusun (nombre N,int b,int n )
   {
   int i = 0;
   if ( verbose ) {
     printf("\n");
     affiche( N, n );
   }
   while ( (i < n) && (N[i] == b-1)) {
        N[i] = 0;
        i = i+1;
   }
   if (i < n)
         N[i] = N[i] + 1;

   if ( verbose ) {
     printf(" -> ");
     affiche( N, n );
     printf(" cout: %d", i);
   }
    return i;
   }
```

2 Makefile

```
inc.pdf : data.pdf inc.tex

inc.exe : inc.c
	gcc -Wall  inc.c -o inc.exe

inc.pdf  : urls.tex data.pdf output.txt inc.tex

	pdflatex inc.tex

output.txt : inc.exe
	echo "\begin{verbatim}"   > output.txt
	./inc.exe -p 2 -b 3 -m 10 >> output.txt
	echo "\end{verbatim}"     >> output.txt

data.txt : inc.exe
	./inc.exe -p 2 -b 2 -m 10 | grep tot | sed 's/t.*=\(.*\)tot.*=\(.*\)/\1 \2/'> b
	./inc.exe -p 2 -b 3 -m 10 | grep tot | sed 's/t.*=\(.*\)tot.*=\(.*\)/\1 \2/'> b
	./inc.exe -p 2 -b 4 -m 10 | grep tot | sed 's/t.*=\(.*\)tot.*=\(.*\)/\1 \2/'> b
	join b2.txt b3.txt > tempo.txt
	join tempo.txt b4.txt > data.txt

plot     :
	echo "set output \"data.fig\"" > gnuplot.cmd
	echo "set term  fig" >> gnuplot.cmd
	echo "set style fill solid 0.75 border -1" >> gnuplot.cmd
	echo "set boxwidth 0.5" >> gnuplot.cmd
	echo "plot 'data.txt' using 1:2 title \"base 2\" w lines, 'data.txt' using 1:3
w lines" >> gnuplot.cmd

data.fig : gnuplot.cmd data.txt
	gnuplot  gnuplot.cmd
data.pdf : data.fig
	fig2dev -Lpdf data.fig > data.pdf
urls.tex    : urls.sh
	./urls.sh > urls.tex
```

La construction du fichier document au format `pdf` dépend de la source latex et d'une image`data.pdf`. L'image est obtenue au moyen de `gnuplot` et d'un fichier de données `data.txt`. Le fichier de commande gnuplot est construit une fois pour toute par `make plot`. Le fichier de données est obtenu par concaténation (commande `join`) de trois exécutions filtrées par les outils standard `grep` et `sed`. Les images sont construites dans un format intermédiaire fig (visualisable et modifiable par le célèbre outil graphique `xfig`) avant d'être

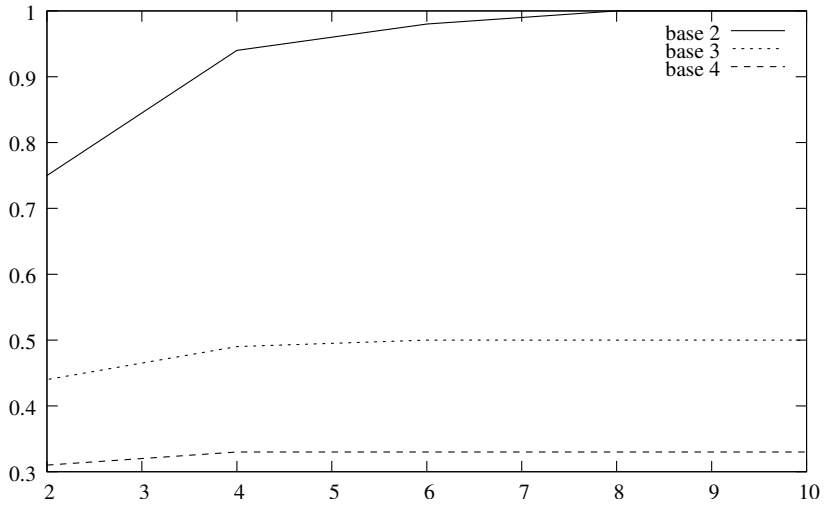

Figure 2: Bases 2, 3, 4, pas 2, max 10

converties au format `pdf` par la commande `fig2dev`.

3 Graphique

4 Liens vers les sources

- makefile.sh urls.sh
- b2.txt b3.txt b4.txt data.txt output.txt tempo.txt
- inc.c
- inc.tex urls.tex
- data.fig
- data.pdf inc.pdf

References

[1] Jean-Pierre Zanotti Travaux-Pratiques d'Algorithmique Module I51, licence. http://zanotti.univ-tln.fr/enseignement/I51/TP2.html

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

5 Listing

```
1  #include<stdio.h>
2  #include<stdlib.h>
3  #include <unistd.h>
4  typedef int* nombre;
5  typedef int chiffre;
6  int verbose = 0;
7  void affiche (nombre z,int max)
8    { int i;
9      for(i= max - 1 ;i >= 0 ;i--)
10        printf(" %d", z[i] );
11    }
12
13 nombre zero (int max)
14   {
15   return (nombre) calloc (max, sizeof( chiffre ));
16   }
17
18 int plusun (nombre N,int b,int n )
19    {
20    int i = 0;
21    if ( verbose ) {
22      printf("\n");
23      affiche( N, n );
24    }
25    while ( (i < n) && (N[i] == b-1)) {
26         N[i] = 0;
27         i = i+1;
28    }
29    if (i < n)
30          N[i] = N[i] + 1;
31
32    if ( verbose ) {
33      printf(" -> ");
34      affiche( N, n );
35      printf(" cout: %d", i);
36    }
37     return i;
38    }
39
40 void output(  int base , int pas , int max)
41 { int nb, cpt, total, qte;
42   nombre N;
43   printf("\nbase=%d max=%d pas=%d", base, max, pas);
44   for ( nb = pas; nb <= max; nb+= pas ) {
45         N = zero ( nb );
46         total = 0;
47         qte = 0;
```

```
48          do {
49             cpt = plusun (N, base, nb);
50             total += cpt;
51             qte ++;
52
53          } while ( cpt < nb );
54          printf("\ntaille=%3d total=%7d moy=%.2f", nb, total, (float) total / qte);
55    }
56  }
57
58  int main (int argc ,char*argv[])
59    {
60    int base=2, pas=1, nb, max=5;
61    int opt;
62    char * optliste = "p:b:m:h:v";
63    while (( opt = getopt( argc, argv, optliste )) >= 0 ) {
64        switch ( opt ) {
65         case 'p' : pas  = atoi(optarg);   break;
66         case 'b' : base = atoi(optarg);   break;
67         case 'm' : max  = atoi(optarg);   break;
68         case 'v' : verbose = 1;           break;
69         case 'h' : printf("\nusage  %s: -p pas -m max -b base -verbose", argv[0]);
70           printf("\ndefaut   : pas=1  max=5 base=2 verbose=0");
71         default  : exit(1);
72        }
73    }
74    output( base, pas, max);
75    printf("\neoj...\n");
76    return 0;
77    }
```