INF203 : Compte-rendu TP10
Encore des automates ...

Alexandre Dupré, Maxime Jaunatre, Clément Raspail | INF - 3
Mail | 13 avril 2021

Syntaxe

Pour ce compte rendu la syntaxe des commandes sera la suivante :

```
[~chemin]: commande
retour de la commande
```

Example :

```
[~/INF203]: ls
sauve_TP1 TP1 TP2
```

Si la commande est interactive et demande d'appuyer sur entrée, une caractère '->' est indiqué. Les fichiers sont en *italique* et les commandes (ou détails de retour de commande) en **gras**. Un script sera donc en gras quand il sera appelé comme une commande.
Les fichiers sources en C sont compilés avec clang, et un nom est donné avec l'option **clang -o**. Cela implique que les programmes seront appelés par un autre nom que **a.out**.

1 Digicode

[a] Le code est **1234**.

1.1 Version programmée

[b] Pour arreter la simulation il faut ecrire **-**.
[c] On test si l'état courant est égale a l'état 4 avec la condition **est_final(etat_courant)** dans la boucle while.

```
[~/INF203/TP10]: clang Digicode_V1/*.c -o digi
[~/INF203/TP10]: ./digi
1
2
3
4
clic
[~/INF203/TP10]: ./digi
1
2
-
```

```
41 [~/INF203/TP10]: tail -n 55 Digicode_V1/automate.c
42 void simule_automate() {
43 int etat_courant, etat_suivant, entree ;
44         entree = 0 ;
```

```
45          etat_courant = etat_initial() ;
46          while (entree != -1 && est_final(etat_courant)) {
47                  entree = lire_entree() ;
48                  switch (etat_courant) {
49                          case Q0 : switch (entree) {
50                                  case 1 : etat_suivant = Q1 ;
51                                          break ;
52                                  default : etat_suivant = Q0 ;
53                                          break ;
54                          }
55                          break ;
56                          case Q1 : switch (entree) {
57                                  case 1 : etat_suivant = Q1 ;
58                                          break ;
59                                  case 2 : etat_suivant = Q2 ;
60                                          break ;
61                                  default : etat_suivant = Q0 ;
62                                          break ;
63                          }
64                          break ;
65                          case Q2 : switch (entree) {
66                                  case 1 : etat_suivant = Q1 ;
67                                          break ;
68                                  case 3 : etat_suivant = Q3 ;
69                                          break ;
70                                  default : etat_suivant = Q0 ;
71                                          break ;
72                          }
73                          break ;
74                          case Q3 : switch (entree) {
75                                  case 1 : etat_suivant = Q1 ;
76                                          break ;
77                                  case 4 : etat_suivant = Q4 ;
78                                          printf("clic\n") ;
79                                          break ;
80                                  default : etat_suivant = Q0 ;
81                                          break ;
82                          }
83                          break ;
84                          case Q4 : switch (entree) {
85                                  case 1 : etat_suivant = Q1 ;
86                                          break ;
87                                  default : etat_suivant = Q0 ;
88                                          break ;
89                          }
90                          break ;
91                          default : break ;
92                  }
93                  etat_courant = etat_suivant ;
94          }
95  }
```

1.2 Version programmée avec fonction

```
[~/INF203/TP10]: clang Digicode_V2/*.c -o digi
[~/INF203/TP10]: ./digi
1234
clic
```

1.3 Version tabulée

[d, e]

```
[~/INF203/TP10]: tail Digicode_V3/automate.c -n36
typedef struct {
  int transitions[NB_ETATS][NB_ENTREES] ;
} transit ;

void init_par_defaut(transit *t) {
  int i, j;
  for (i = 0; i < NB_ETATS; i++)
    for (j = 0; j < NB_ENTREES; j++)
      t->transitions[i][j] = Q0;
}

void transition(transit *t){
  init_par_defaut(t);
  t->transitions[Q0][1] = Q1;
  t->transitions[Q1][1] = Q1;
  t->transitions[Q2][1] = Q1;
  t->transitions[Q3][1] = Q1;
  t->transitions[Q4][1] = Q1;
  t->transitions[Q1][2] = Q2;
  t->transitions[Q2][3] = Q3;
  t->transitions[Q3][4] = Q4;
}

void simule_automate() {
  int etat_courant, etat_suivant, entree ;
  transit t;
  transition(&t);
  entree = 0 ;
  etat_courant = etat_initial() ;
  while (entree != -1 && est_final(etat_courant)) {
    entree = lire_entree() ;
    etat_courant = t.transitions[etat_courant][entree] ;
    if(etat_courant == Q4)
      printf("clic\n");
  }
}
```

[f] Le code peut comporter plusieurs fois le meme numéro mais étant donné qu'a chaque fois qu'on double le numéro, le code final sera toujours de la forme *1234, on peut donc en déduire que seul le code final compte.
Exemple : **1234**, 111**1234**, 12312**1234**.
[g] L'inconvénient c'est qu'on ne peut pas répéter un numero dans le code sans devoir modifier la structure entière de l'automate. L'avantage de la version 2 par rapport à la 1 est que le switch

étant dans une fonction séparée, il est plus facile de le modifier et est plus lisible (compréhensible par rapport à s'il était dans une autre fonction). La version 3 est la plus optimale, pour rajouter un cas il suffit juste de rentrer dans le tableau la valeur de transition alors que pour un switch il faudrait rajouter plusieurs lignes pour vérifier cette condition.

1.4 Version tabulée

[h]

		Entrées					
		0	1	2	3	4	5-9
États suivants	0	0	**1**	0	0	0	0
	1	0	**1**	**2**	0	0	0
	2	0	**1**	0	**3**	0	0
	3	0	**1**	0	0	**4**	0
	4	0	**1**	0	0	0	0

TABLE 1 – Tableau de transition de l'automate Digicode

On peut voir que le code de la porte est inscrit dans la diagonale. Chaque chiffres correspond à la transition n = n+1 sauf 4 car il est la fin, il fini donc la boucle.
[i] On a utilisé le format .txt pour stocker les valeurs comme dans le tp précédant.

```
[~/INF203/TP10]: cat auto.txt
8
0 1 1
1 1 1
2 1 1
3 1 1
4 1 1
1 2 2
2 3 3
3 4 4
```

```
43 [~/INF203/TP10]: tail -n 30 Digicode_V4/automate.c | head -n 18
44 int init_automate(char* nom, transit *t){
45   int nb_trans, dep, e, arr, i;
46   FILE *f;
47   f = fopen(nom,"r");
48   if (f == NULL){
49     return 1;
50   }
51   init_par_defaut(t);
52
53   fscanf(f, "%d", &nb_trans);
54   for (i=1 ; i<= nb_trans ; i++) {
55     fscanf(f, "%d %d %d", &dep, &e, &arr);
56     t->transitions[dep][e] = arr ;
57   }
58   fclose(f);
59   return 0;
60 }
```