Algorithmes sur les permutations

Cours du Dea Informatique Labri Septembre 1997

1 Géneration de toutes les permutations

Il s'agit d'afficher toutes les permutations sur n élements. L'algorithme suivant donne des résultats satisfaisants jusqu'à $n = 11$. La permutation `alpha` est une variable globale:

```
int alpha[Max+1];

void Echange (int i , int j){
    int temp;
    temp = alpha[i];
    alpha[i] = alpha[j];
    alpha[j] = temp;
}


void GenerePerm (int n, int i){
    int j;
    if (i > n) Affiche(n, alpha);
    else
         for (j = 1; j <= i; ++j){
            Echange(i,j);
            GenerePerm(n,i+1);
            Echange(j,i);
   }
}
```

Il faut appeler cette procédure dans une procédure principale:

```
void main() {
   int i,n;
   Lire(&n);
   for (i =1; i<= n; ++i) alpha[i] = i;
   GenerePerm(n, 1);
}
```

2 Bijection entre tableaux d'inversions et permutations

Rappel de définition: Etant donnée une permutation $\alpha = a_1, a_2, \ldots, a_n$ une inversion est un couple a_i, a_j tel que $i < j$ et $a_i > a_j$. On construit le tableau t_α tel que $t_\alpha[k]$ est le nombre d'inversions de la forme (i, k). A partir d'un tableau t_α satisfaisant la condition:

$$\forall i, 1 \leq i \leq n \qquad \text{on a} \qquad 0 \leq t[i] \leq n - i$$

on construit une unique permutation α telle que $t = t_\alpha$ par l'algorithme:

```
void TransformTabPerm (int t[], int alpha[], int n){
     int i, j,  cpt;
     for (i = 1; i <= n; ++i) alpha[i] = 0;
     for (i = 1; i<= n; ++i){
         cpt = 0;
         for (j = 1; cpt <= t[i]; ++j)
           if (alpha[j] == 0) {
               cpt++;
               if (cpt> t[i]) alpha[j] = i;
               }
     }
 }
```

Ou par celui-ci:

```
void TransformTabPerm (int t[], int alpha[], int n){
     int j;
     for (i = n; i>= 1 n; --i){
         for (j = n-i; j > t[i]+1 ; --j)
             alpha[j] = alpha[j-1];
         alpha[j] = i;
      }
 }
```

3 Génération aléatoire de permutations

Le plus simple est d'engendrer aléatoirement un tableau d'inversions puis de le transformer en permutation:

```
void HasardPerm (int n){
     int t[Max +1];
     int i, u;
     for (i =1; i < n; ++i)
        t[i] = rand()%(n-i);
     t[n] = 0;
     TransformTabPerm(t,alpha,n);
    }
```