Cours 11 : Rappel sur les ABR, Parcours et Rotations

Algorithmique et programmation procédurale

Rappel sur les arbres

- Rappel de quelques notions clés
  - Sommet
  - Racine
  - Père
  - Fils
  - Branche
  - Feuille

Rappel sur les arbres

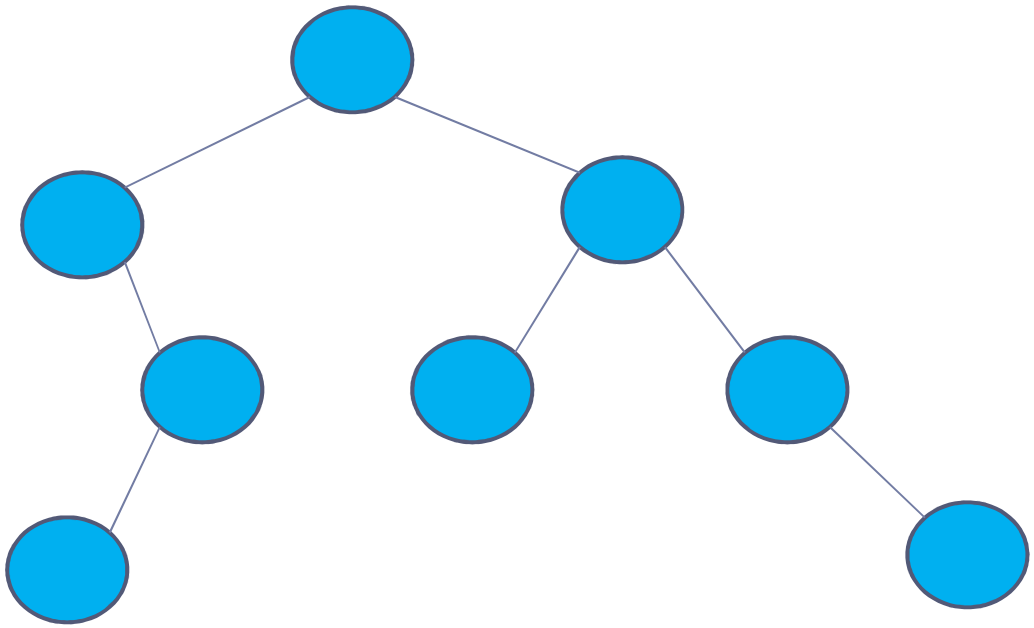
- Rappel de quelques notions clés
  - Degré de branchement
  - Taille
  - Hauteur
  - Longueur de cheminement

Rappel sur les arbres

- Rappel de quelques notions clés
  - Filiforme
  - Complet
  - Parfait

Rappel sur les arbres binaires

- Arbre binaire
  - Nombre maximal de fils d'un nœud est 2



Pseudo-code des arbres binaires

- Constante
  - arbreVide

- Fonctions
  - **estVide** : ArbreBinaire -> Booleen
  - **cons** : T × ArbreBinaire × ArbreBinaire -> ArbreBinaire
  - **racine**, **r** : ArbreBinaire -> T
  - **filsGauche**, **fG** : ArbreBinaire -> ArbreBinaire
  - **filsDroit**, **fD** : ArbreBinaire -> ArbreBinaire

Exercice

- Ecrire une procédure qui affiche le parcours en profondeur (infixe) d'un arbre binaire

Exercice

```
Procédure parcoursProfondeur(A: ArbreBinaire)
Début
  Si estVide (A) Alors
        Retourner
  Sinon
        parcoursProfondeur(filsGauche(A))
        Ecrire racine(A)
        parcoursProfondeur(filsDroit(A))
  FinSi
Fin
```

Parcours d'un arbre

- Parcours en profondeur
  - Visiter les fils avant les frères
  - Algorithme récursif
  - Algorithme itératif utilisant une pile
- Parcours en largeur
  - Visiter les frères avant les fils
  - Algorithme itératif utilisant une file

Exercice

- Ecrire une procédure qui affiche le parcours en largeur d'un arbre binaire

```
Procédure parcoursLargeur(A : ArbreBinaire)
Variables f : File, ab : ArbreBinaire
Début
   f <- fileVide()
   Si !estVide(A) Alors enfiler(f, A)
   Tantque ! estVide(f)
         ab <- defiler(f)
         Ecrire racine(ab)
         Si !estVide(fG(ab)) Alors enfiler(f, fG(ab)) Finsi
         Si !estVide(fD(ab)) Alors enfiler(f, fD(ab)) Finsi
   FinTantque
Fin
```

Les arbres binaires en C

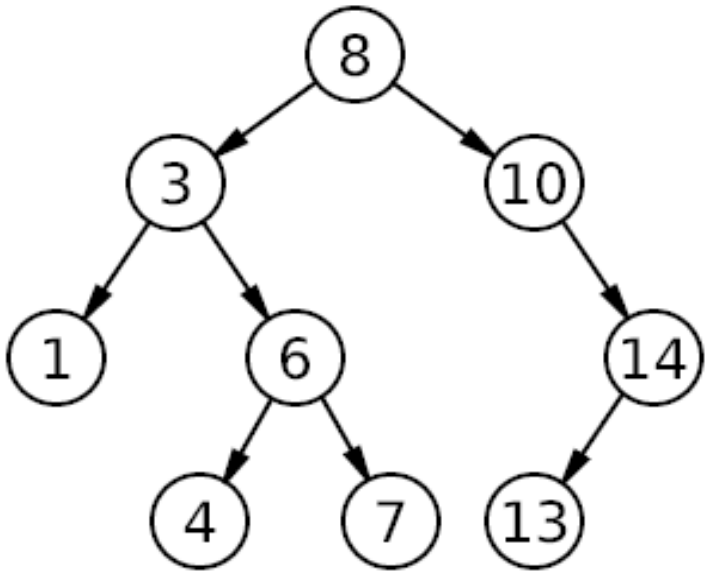
```
typedef struct Sommet {
    int racine;
    struct Sommet* filsG;
    struct Sommet* filsD;
} Sommet;
```

Exemple en C

```
int hauteur(Sommet * a) {
  if (a == NULL)
        return -1;
  if (a->filsG == NULL  &&  a->filsD == NULL)
        return 0;
  return 1+max(hauteur(a->fG), hauteur(a->fD));
}
```

Rappel sur les ABR

- Arbre binaire de recherche
  - À chaque sommet de l'arbre
    - tous les éléments du sous-arbre gauche ont une valeur de clé inférieure à celle du nœud courant, et
    - tous les éléments du sous-arbre droit ont une valeur de clé supérieure



Rappel sur les ABR

- Opérations
  - Recherche
  - Ajout en feuille
  - Ajout à la racine
  - Suppression

Rappel sur les ABR

Ajout en feuille

```
Fonction insérer(x: T, A: ABR): ABR
Début
  Si estVide(A) Alors
       Retourner cons(x,arbreVide, arbreVide)
  SinonSi x=r(A)
       Retourner A        // ne rien faire
  SinonSi x>r(A)
       Retourner cons(r(A),fG(A), insérer(x,fD(A)))
  Sinon {x<r(A)}
       Retourner cons(r(A),insérer(x,fG(A)), fD(A))
  FinSi
Fin
```

Exercice

- Ecrire la fonction C de l'ajout en feuille

Exercice

```
Sommet* inserer (int n, Sommet* sommet) {
  if (sommet == NULL) {
       Sommet* s = (Sommet*)malloc(sizeof(Sommet));
        s->racine = n;
        s->filsG = NULL;
        s->filsD = NULL;
        return s;
  }
  else if (sommet->racine<n)
       sommet->filsD = inserer (n, sommet->filsD);
  else if (sommet->racine>n)
       sommet->filsG = inserer (n, sommet->filsG);
  return sommet;
}
```

Complexité dans un ABR

- Recherche, insertion, suppression
  - O(h) où h est la hauteur de l'arbre
  - Pire cas : arbre filiforme (dégénéré)
  - Meilleur cas : arbre complet ou *arbre équilibré*

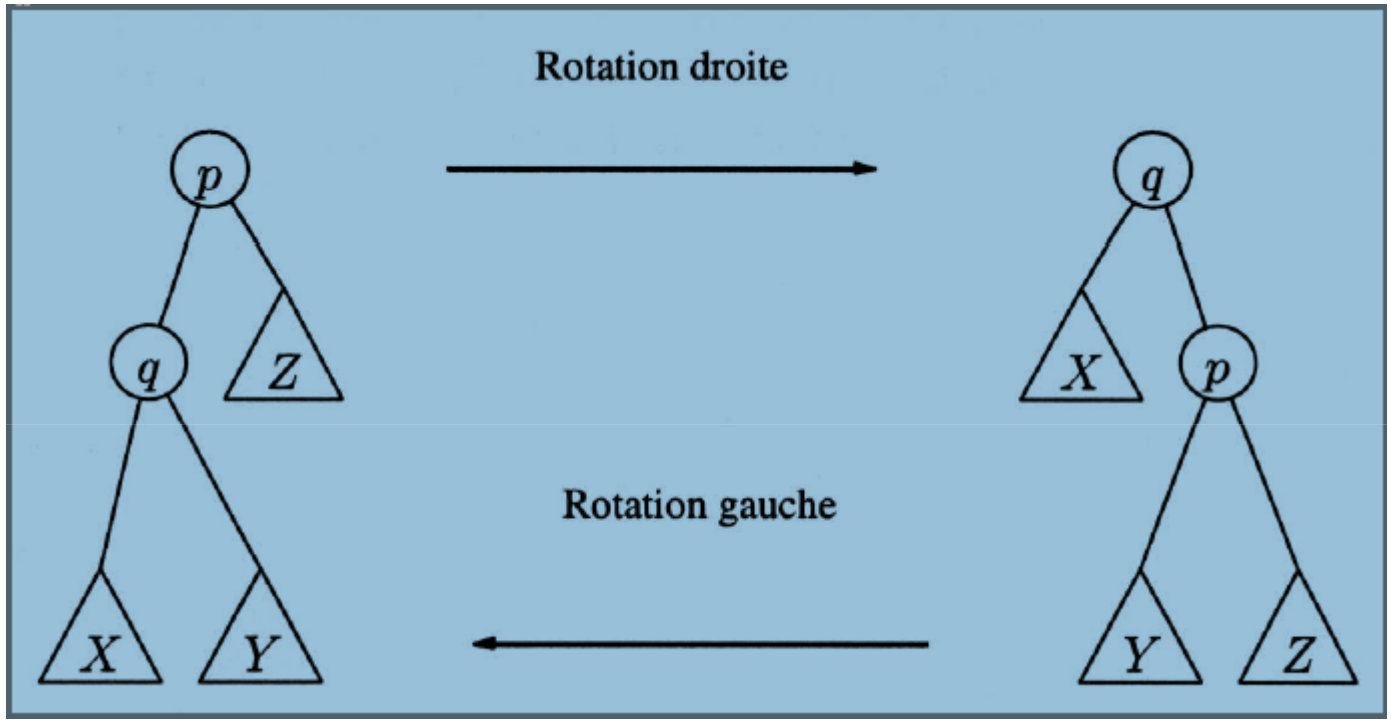
Rotations

- Définition
  - Un arbre binaire est équilibré si pour tous les sommets, la hauteur du fils gauche reste voisine de celle du fils droit, c'est-à-dire que

    |hauteur(filsGauche) - hauteur(filsDroit)| <= 1

- But
  - Rééquilibrer les arbres après les opérations d'insertion et de suppression.

Rotations simples

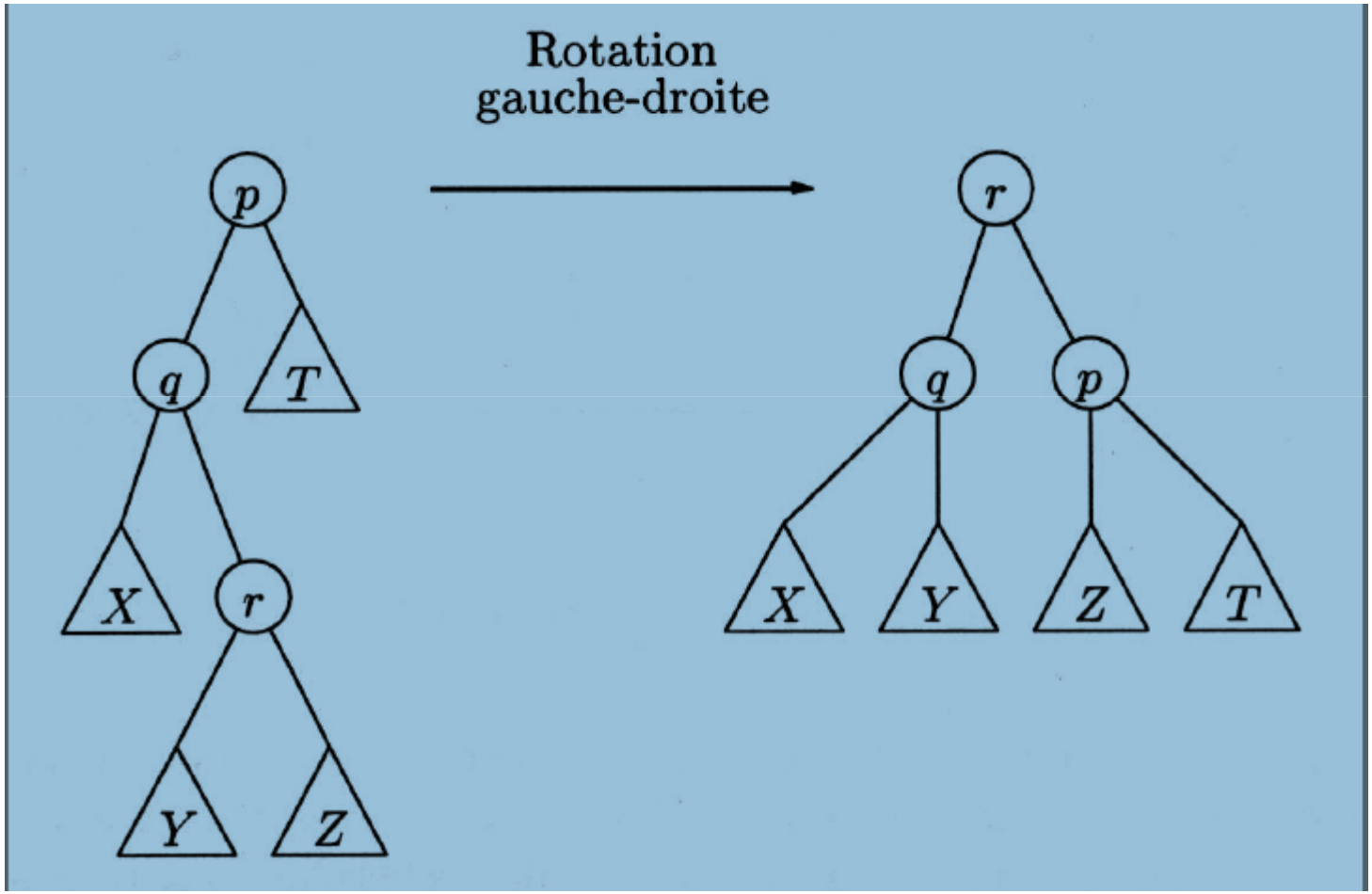


- Si A est un ABR, l'arbre obtenu en appliquant une rotation simple à l'un de ses sommets est un ABR.

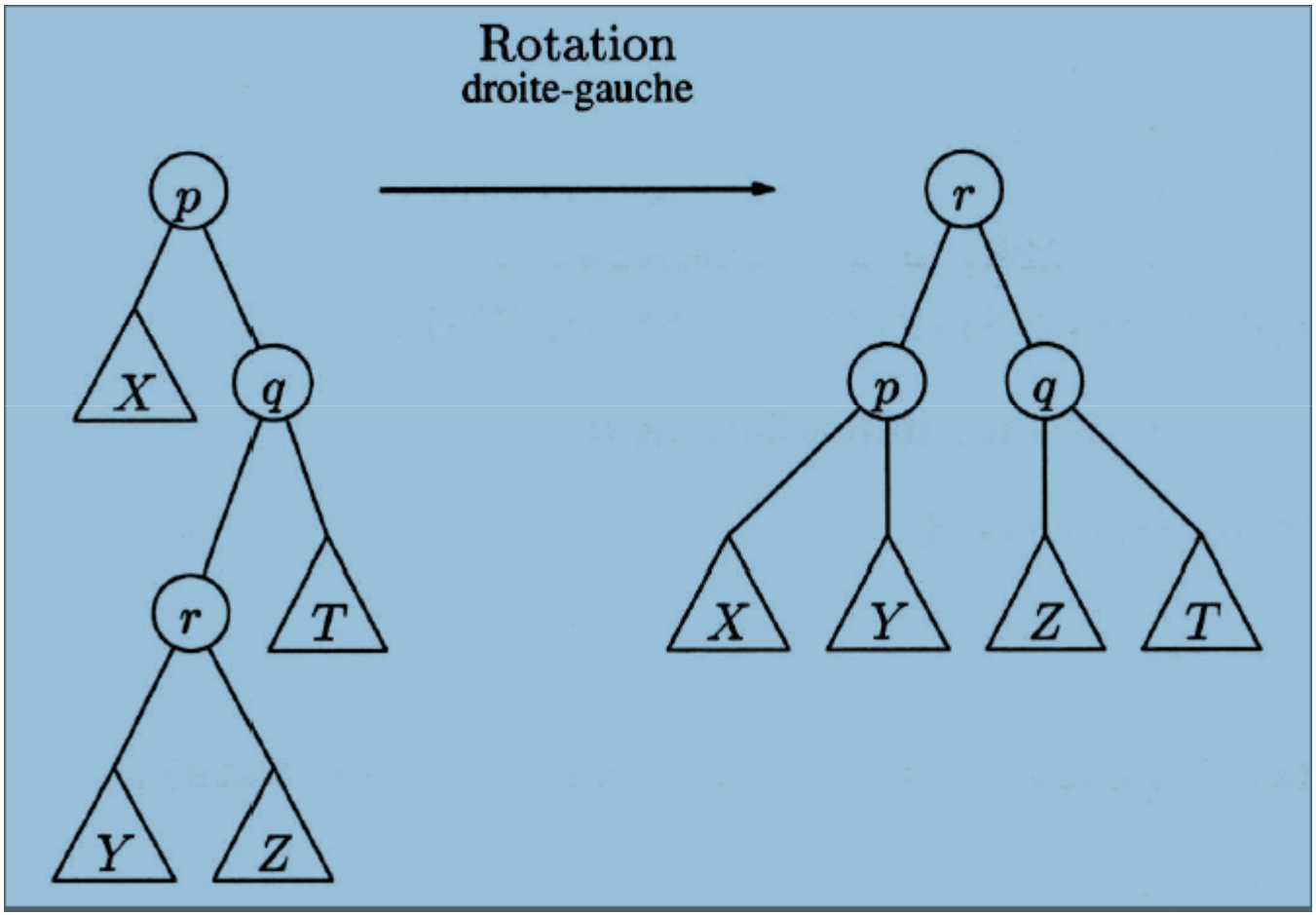
Rotations doubles

- Rotation gauche-droite :
  - D'abord une rotation gauche sur le sous-arbre gauche
  - Puis une rotation droite sur l'arbre résultant
- Rotation droite-gauche :
  - D'abord une rotation droite sur le sous-arbre droit
  - Puis une rotation gauche sur l'arbre résultant

Rotation gauche-droite



Rotation droite-gauche



Exercice

- Quelles rotations sont nécessaires pour équilibrer cet ABR?