

Cours n° 5

Structures de données abstraites



Sommaire

1. Structures linéaires

- 1.1 Pile
- 1.2 File
- 1.3 Deque
- 1.4 Liste
- 1.5 File à priorité
- 1.6 Liste triée

2. Structures de graphe

- 2.1 Implémentation d'un graphe
- 2.2 Parcours dans un graphe

Composition de données unies par une même sémantique

Sémantique d'une structure de données

Ensemble d'opérations abstraites définies sur les données

- Préconditions sur ces opérations et axiomatique

Utilisation de types abstraits de données (entiers, réels, ...)

Implémentation d'une structure de données

Choix d'une structure concrète dans un langage donnée (C++, Java, ...),

Critères de la réalisation

- Simplicité de l'implémentation
- Efficacité de l'accès aux données

Bibliographie

C. Carrez , «Structures de données en Java, C++ et Ada», Dunod.

www.cert.fr/dcsd/cd/MEMBRES/lemaitre/Enseignement/SA/polySA.pdf

www.nist.gov/dads (Dictionary of Algorithms and Data Structures)

Définition

Séquence d'éléments

$S = \langle e_1 e_2 e_3 \dots e_n \rangle$ avec cardinalité(S) = n ,

Opération successeur définie (précondition : sauf sur le dernier élément),

Type d'accès à un élément (séquentiel, direct)

Existence ou non d'une relation d'ordre entre éléments

Pile, File et Dèque (accès séquentiel à un élément, pas de relation d'ordre)

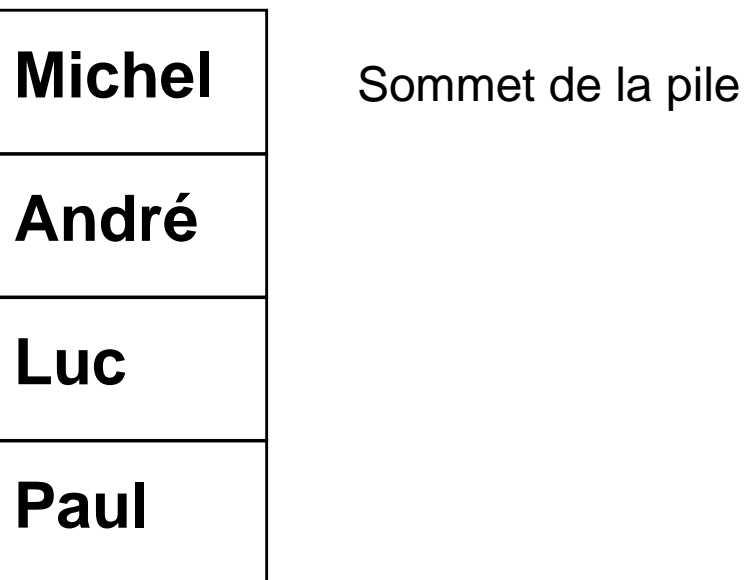
File à priorité (accès séquentiel à un élément, relation d'ordre)

Liste (accès direct à un élément), **Liste triée** (relation d'ordre)

Introduction

Séquence d'éléments accessibles par une seule extrémité appelée sommet

Modèle dernier entré – premier sorti (Last In – First Out) LIFO



Pile de quatre noms (P_n)

Définition abstraite

Type pile

Paramètre élément **utilise** booléen

Opérations

Pilevide : \rightarrow pile

Création d'une pile vide

Estvide : pile \rightarrow booléen

Est-ce que la pile est vide ?

Sommet : pile \rightarrow élément

Lecture du sommet d'une pile

Empiler : pile x élément \rightarrow pile

Ajout d'un élément en sommet de pile

Dépiler : pile \rightarrow pile

Suppression du sommet de pile

Préconditions (pile p, élément e)

Sommet(p) : \neg Estvide(p)

Dépiler(p) : \neg Estvide(p)

Sémantique (pile p, élément e)

Estvide(Pilevide),

\neg Estvide(Empiler(p,e)),

Sommet(Empiler(p,e)) := e

Dépiler(Empiler(p,e)) := p

Exemple

Création de la pile Pn

$P_n := \text{Pilevide}$

$\text{Empiler}(P_n, \text{Paul}) \text{ Empiler}(P_n, \text{Luc}) \text{ Empiler}(P_n, \text{André}) \text{ Empiler}(P_n, \text{Michel})$

Suite d'actions

$\text{Dépiler}(P_n) \text{ Empiler}(P_n, \text{Jean}) \text{ Dépiler}(P_n) \text{ Dépiler}(P_n) e := \text{Sommet}(P_n)$

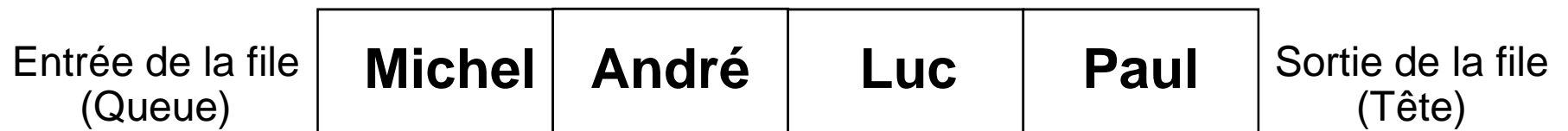
Question 1) Quelle est la valeur de l'élément e ?

Question 2) Dessiner la pile Pn

Introduction

**Séquence d'éléments accessibles par deux extrémités
appelées Tête et Queue (une en lecture, une en écriture)**

Modèle premier entré – premier sorti (First In – First Out) FIFO



File de quatre noms (Fn)

Définition abstraite

Type file

Paramètre élément **utilise** booléen

Opérations

Filevide : -> file

Création d'une file vide

Estvide : file -> booléen

Est-ce que la file est vide ?

Tête : file -> élément

Lecture de l'élément en tête de file

Ajouter : file x élément -> file

Ajout d'un élément en queue de file

Retirer : file -> file

Suppression de l'élément en tête de file

Préconditions (file f, élément e)

Tête(f) : \neg Estvide(f)

Retirer(f) : \neg Estvide(f)

Sémantique (file f, élément e)

Estvide(Filevide),

\neg Estvide(Ajouter(f,e)),

Tête(Ajouter(f,e)) := si Estvide(f) alors e sinon tête(f) fsi

Retirer(Ajouter(f,e)) := si Estvide(f) alors Filevide sinon Ajouter(Retirer(f), e) fsi

Exemple

Création de la file Fn

$\text{Fn} := \text{Filevide}$

Ajouter(Fn, Paul) Ajouter(Fn, Luc) Ajouter(Fn, André) Ajouter(Fn, Michel)

Suite d'actions

Retirer(Fn) Ajouter (Fn, Jean) Retirer(Fn) Retirer(Fn) $e := \text{Tête}(\text{Fn})$

Question 1) Quelle est la valeur de l'élément e ?

Question 2) Dessiner la file Fn

Introduction

**Séquence d'éléments accessibles par deux extrémités
appelées Gauche et Droite (en lecture et en écriture)**

Propriétés cumulés d'une pile et d'une file

Entrée/sortie
gauche
de la dèque

Michel	André	Luc	Paul
---------------	--------------	------------	-------------

Entrée/sortie
droite
de la dèque

Dèque de quatre noms (Dn)

Définition abstraite

Type dèque

Paramètre élément **utilise** booléen

Opérations

Dèquevide : -> dèque

Estvide : dèque -> booléen

Extrémité : dèque x sens -> élément

Ajouter : dèque x élément x sens -> dèque

Retirer : dèque x sens -> dèque

Création d'une dèque vide

Est-ce que la dèque est vide ?

Lecture de l'élément d'une l'extrémité

Ajout d'un élément à une extrémité

Suppression d'un l'élément à une extrémité

Préconditions (dèque d, élément e, sens s)

Extrémité(d, s) : $\neg \text{Estvide}(d)$

Retirer(d, s) : $\neg \text{Estvide}(d)$

Sémantique (dèque d, élément e, sens s)

Estvide(Dèquevide),

$\neg \text{Estvide}(\text{Ajouter}(d,e,s)),$

Extrémité(Ajouter(d,e, s), $\neg s$) := si Estvide(d) alors e sinon extrémité(d, $\neg s$) fsi

Retirer(Ajouter(d,e, s), $\neg s$) := si Estvide(d) alors Dèquevide

sinon Ajouter(Retirer(d, $\neg s$), s) fsi

Retirer(Ajouter(d,e, s), s) := d

Exemple

Création de la dèque Dn

Dn := Dèquevide

Ajouter(Dn, André, gauche) Ajouter(Dn, Luc, droite)

Ajouter(Dn, Michel, gauche) Ajouter(Dn, Paul, droite)

Suite d'actions

Retirer(Dn, gauche) Ajouter (Dn, Jean, droite) Retirer(Dn, gauche)

Retirer(Dn, droite) e := Extrémité(Dn, gauche)

Question 1) Quelle est la valeur de l'élément e ?

Question 2) Dessiner la dèque Dn

Introduction

Séquence d'éléments accessibles par leur indice

Accès direct à un élément

Eléments	Michel	André	Luc	Paul
Indice	1	2	3	4

Liste de quatre noms (Ln)

Définition abstraite

Type liste

Paramètre élément, entier positif (indice) **utilise** entier positif

Opérations

Listevide : \rightarrow liste

Création d'une liste vide

Longueur : liste \rightarrow entier

Nombre d'éléments de la liste

Lire : liste x entier \rightarrow élément

Lecture de l'élément d'indice donné

Insérer : liste x élément x entier \rightarrow liste

Insertion d'un élément à un indice donné

Supprimer : liste x entier \rightarrow liste

Suppression de l'élément d'indice donné

Préconditions (liste l, élément e, entier i)

Lire(l, i) : $\neg(\text{Longueur}(l) < i)$

Supprimer(l, i) : $\neg(\text{Longueur}(l) < i)$, Insérer(l, e, i) : $\neg(\text{Longueur}(l) < i-1)$,

Sémantique (liste l, élément e, entiers i et j)

Longueur(Listevide) := 0,

Longueur(Insérer(l,e,i) := Longueur(l)+1, Longueur(Supprimer(l,e,i) := Longueur(l)-1,

Lire(Insérer(l,e,i), j) := si (j < i) alors Lire(l, j) sinon Lire(l,j-1)

Lire(Supprimer(l,i), j) := si (j < i) alors Lire(l, j) sinon Lire(l,j+1)

Supprimer(Insérer(l, e, i), i) := l

Exemple

Création de la file Ln

$Ln := \text{Listevide}$

Insérer(Ln, André, 1) Insérer(Ln, Michel, 1)

Insérer(Ln, Luc, 3) Insérer(Ln, Paul, 4)

Suite d'actions

Supprimer(Ln, 1) Insérer(Ln, Jean, 2) Supprimer(Ln, 1)

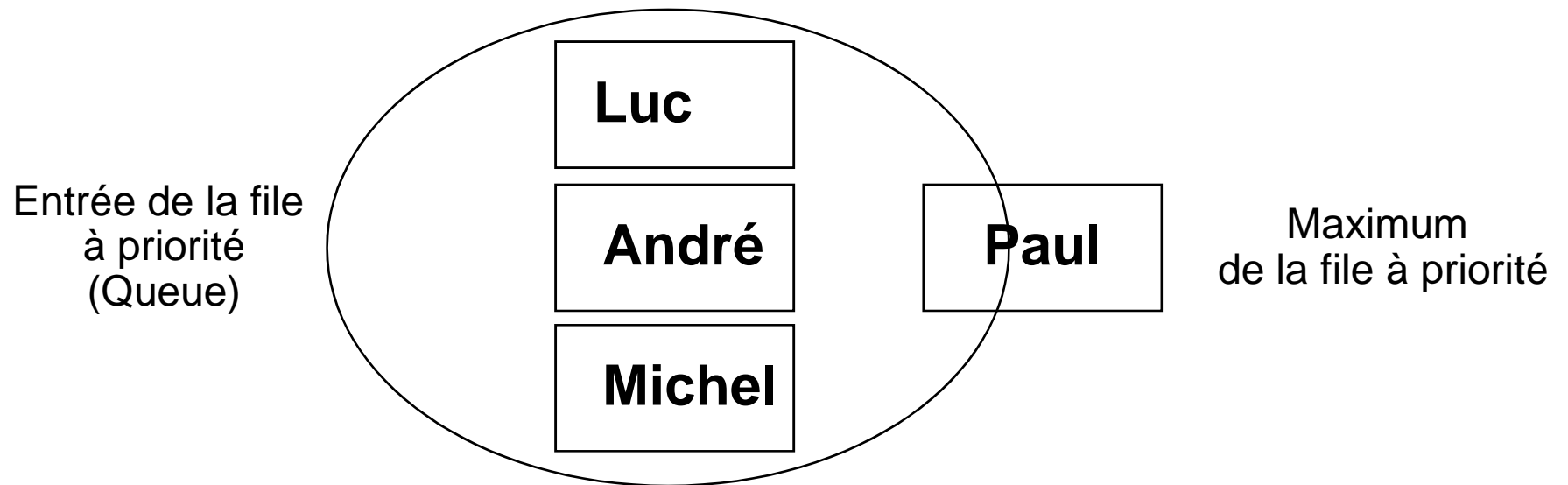
Supprimer(Ln, 2) $e := \text{Lire}(1)$

Question 1) Quelle est la valeur de l'élément e ?

Question 2) Dessiner la liste Ln

Introduction

Structure de File + relation d'ordre entre les éléments



File à priorité de quatre noms (FPn)

Ordre lexicographique

Définition abstraite (1/2)

Type file à priorité

Paramètre élément **utilise** booléen

Opérations

Filevide : -> file à priorité

Création d'une file vide

Estvide : file à priorité -> booléen

Est-ce que la file est vide ?

Ordre : élément x élément -> booléen

Est-ce que les deux éléments sont dans le bon ordre?

Tête : file à priorité -> élément

Lecture de l'élément d'ordre maximal

Ajouter : file à priorité x élément -> file à priorité

Ajout d'un élément en queue de file

Retirer : file à priorité -> file à priorité

Suppression de l'élément de rang maximum

Préconditions (file à priorité fp, élément e)

Tête(fp) : \neg Estvide(fp),

Retirer(fp) : \neg Estvide(fp)

Définition abstraite (2/2)

Sémantique (file à priorité fp, élément e)

Estvide(Filevide),

\neg Estvide(Ajouter(fp,e)),

Tête(Ajouter(fp,e)) := si Estvide(fp) alors e sinon
si Ordre(Maximum(fp), e) alors e sinon Tête(fp) fsi fsi

Retirer(Ajouter(fp,e) := si Estvide(fp) alors Filevide sinon
si Ordre(Tête(fp), e) alors fp sinon Ajouter(Retirer(fp), e) fsi fsi

Exemple

Création de la file FPn

FPn := Filevide

Ajouter(FPn, Paul) Ajouter(FPn, André) Ajouter(FPn, Luc) Ajouter(FPn, Michel)

Suite d'actions

Retirer(FPn) Ajouter (FPn, Jean) Retirer(FPn) Retirer(FPn) e := Maximum(FPn)

Question 1) Quelle est la valeur de l'élément e ?

Question 2) Dessiner la file FPn

Définition abstraite (1)

Type liste triée (extension liste)

Paramètre élément, entier positif (indice) **utilise** entier positif

Opérations

Ordre : élément x élément \rightarrow booléen

Premier : liste \rightarrow élément

Dernier : liste \rightarrow élément

ListeInf : liste x entier \rightarrow liste

ListeSup : liste x entier \rightarrow liste

EstTriée : liste \rightarrow booléen

EstTriéel : liste \rightarrow booléen

Relation d'ordre entre éléments

Premier élément de la liste

Dernier élément de la liste

Liste des i premiers éléments

Liste des i derniers éléments

Est-ce une liste triée dans l'ordre?

Est-ce une liste triée dans l'ordre inverse?

Sémantique (liste l, élément e, entiers i et j)

Premier(l) := Lire(l, 0),

Dernier(l) := Lire(l, Longueur(l)-1)

ListeInf(l, i) := si (longueur(l) \leq i) alors l sinon ListeInf(Supprimer(l, i))

ListeSup(l, i) := si (longueur(l) \leq i) alors l sinon ListeInf(Supprimer(l, 0))

Définition abstraite (2)

EstTriée (l) := si (Longueur(l) = 0) alors vrai
 sinon (Ordre(premier(l), premier(ListeSup(Longueur(l)-1)))
 et EstTriée(ListeSup(Longueur(l)-1)))

EstTriéel(l) := si (Longueur(l) = 0) alors vrai
 sinon (Ordre(premier(ListeSup(Longueur(l)-1)), premier(l))
 et EstTriéel(ListeSup(Longueur(l)-1)))

Définitions

Ensemble X d'éléments (sommets) reliés deux à deux par des arcs valués

$u_{(x, y)}$ arc du sommet x au sommet y avec comme valeur $v(u_{(x, y)})$

$G = \langle X, U \rangle$ avec U ensemble des arcs,

$\text{ordre}(G) = \text{cardinalité}(X) = n, \text{cardinalité}(U) \leq n^2$

Graphe non orienté

si $u_{(x, y)}$ existe, alors $u_{(y, x)}$ existe et $v(u_{(x, y)}) = v(u_{(y, x)})$

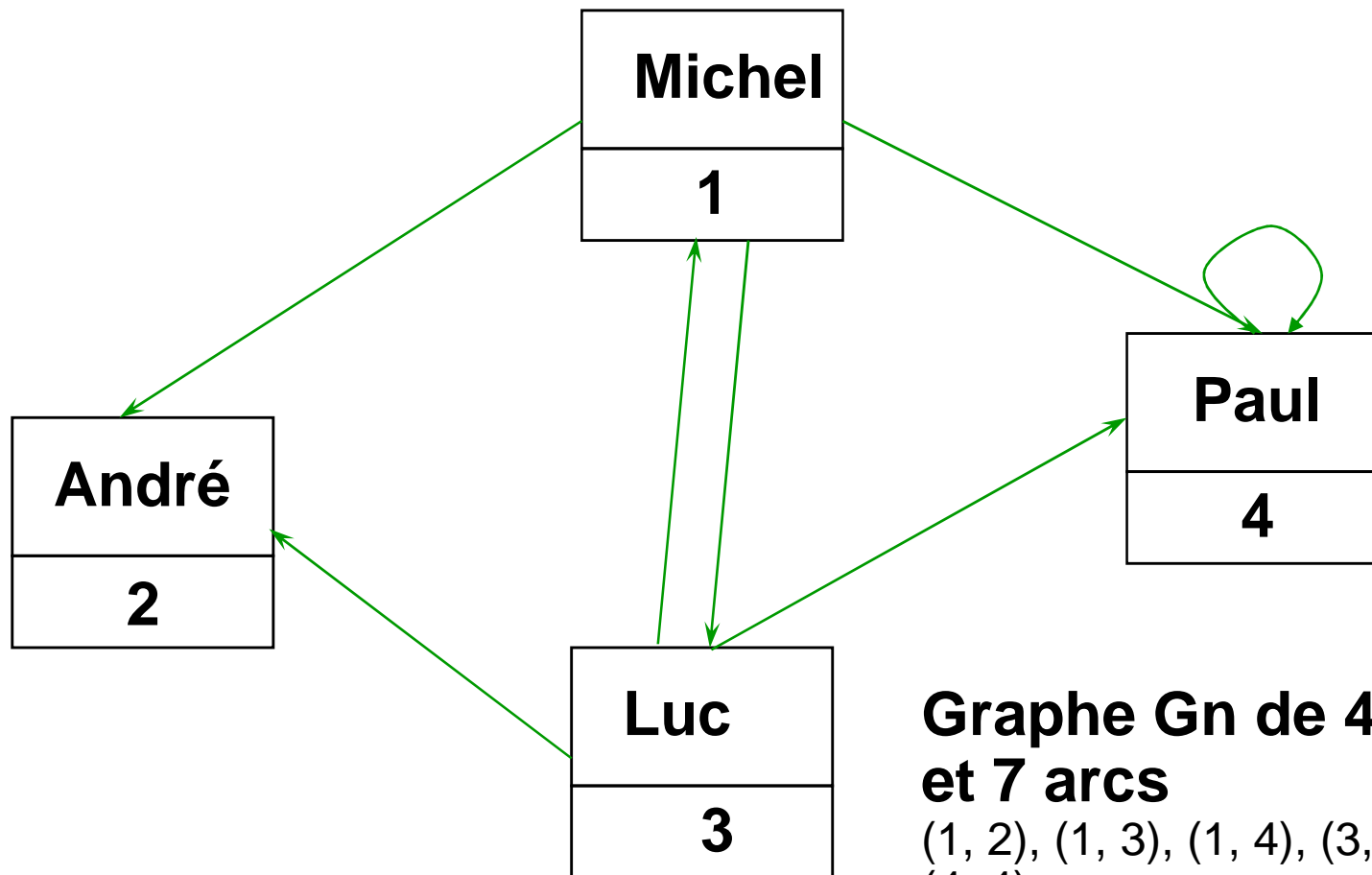
Arbre

Sommet particulier appelé racine,

Chemin (suite d'arcs) unique de la racine jusqu'à chaque sommet,

Graphe connexe sans cycle

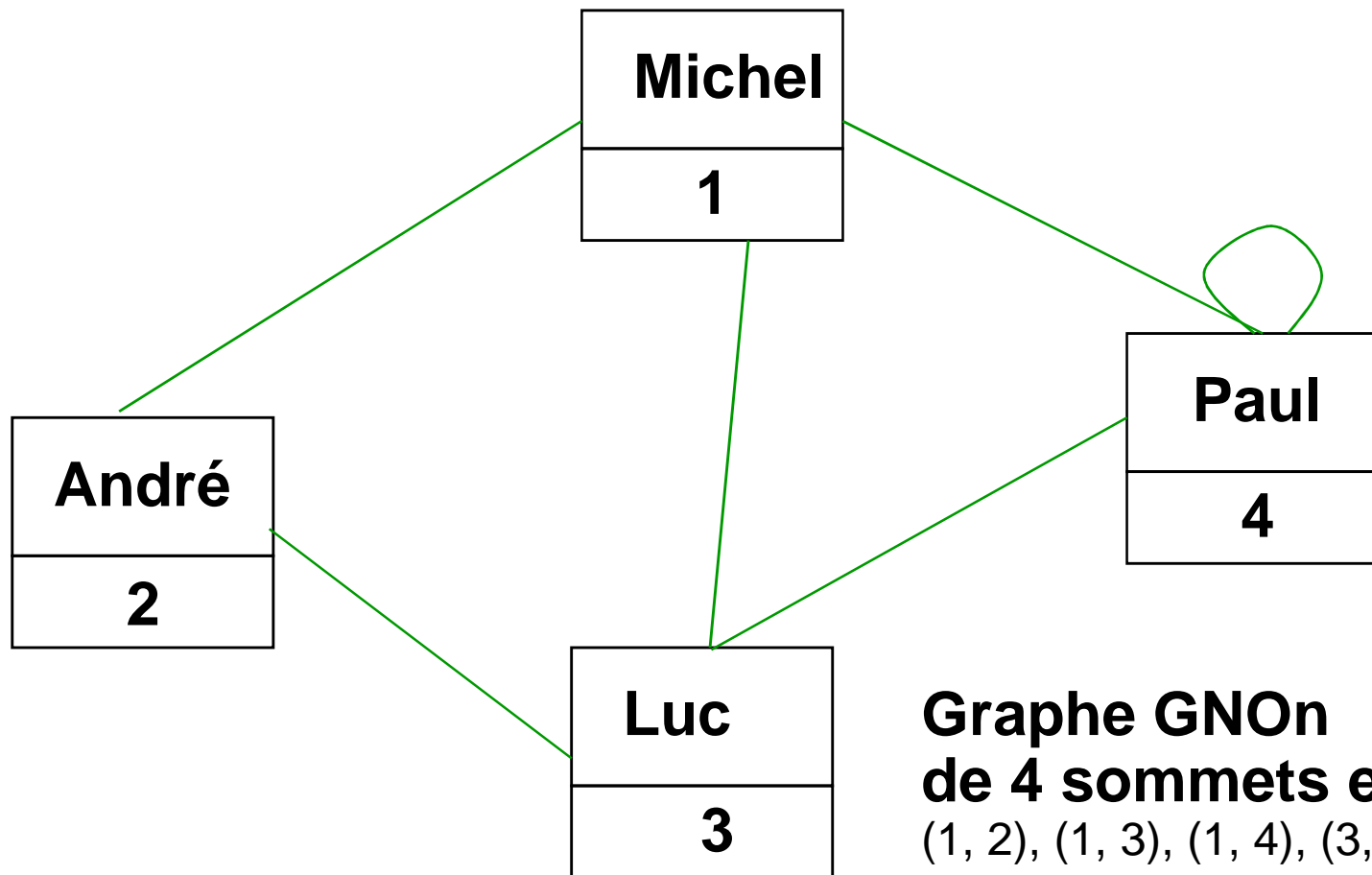
Exemple de graphe orienté



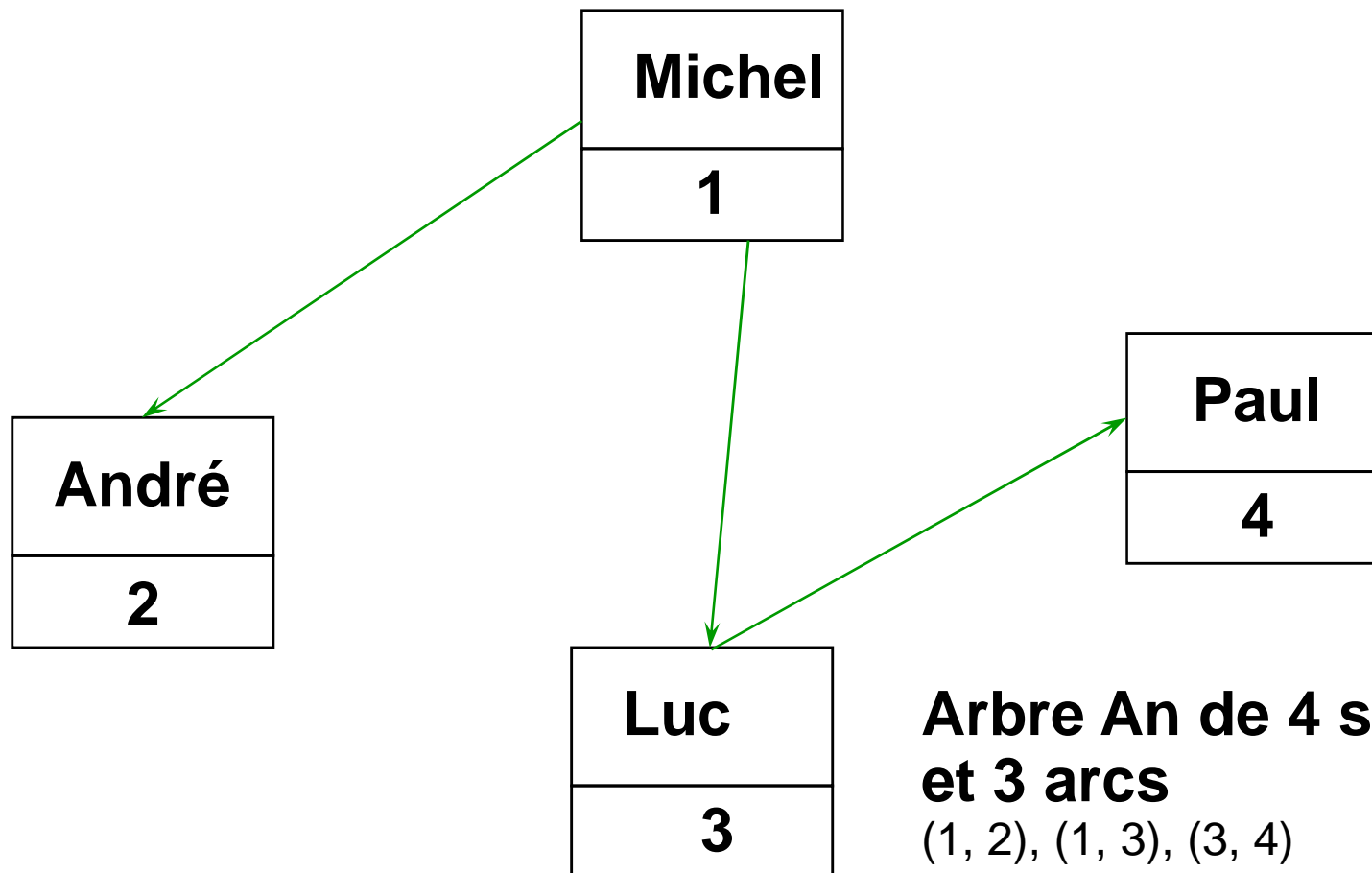
**Graphe G_n de 4 sommets
et 7 arcs**

$(1, 2), (1, 3), (1, 4), (3, 1), (3, 2), (3, 4),$
 $(4, 4)$

Exemple de graphe non-orienté



Exemple d'arbre



**Arbre An de 4 sommets
et 3 arcs**

(1, 2), (1, 3), (3, 4)

2. STRUCTURES DE GRAPHS

Définition abstraite

Type graphe

Paramètre sommet **utilise** booléen

Opérations

Graphevide : \rightarrow graphe vide

Création d'un graphe vide

Ordre : graphe \rightarrow entier

Nombre de sommets

Arc : graphe x sommet x sommet \rightarrow booléen

Existence d'un arc

d+ : graphe x sommet \rightarrow entier

Nombre d'arcs partant d'un sommet

d- : graphe x sommet \rightarrow entier

Nombre d'arcs arrivant à un sommet

AjouterArc : graphe x sommet x sommet \rightarrow graphe

SupprimerArc : graphe x sommet x sommet \rightarrow graphe

AjouterSommet : graphe x sommet \rightarrow graphe

SupprimerSommet : graphe x sommet \rightarrow graphe

Sémantique (graphe g, sommets s1 et s2, entier i)

Ordre(Graphevide) := 0,

Exemple

Création de l'arbre An

An := Graphevide

AjouterSommet (An, Michel), AjouterSommet (An, André),
AjouterSommet (An, Luc), AjouterSommet (An, Paul),
AjouterArc (An, Michel, André), AjouterArc (An, Michel, Luc),
AjouterArc (An, Luc, Paul),

Suite d'actions

SupprimerSommet(An, Luc), AjouterSommet (An, Jean),
SupprimerSommet(An, Paul), AjouterArc (An, Michel, Jean),
b := Arc(Luc, Paul)

Question 1) Quelle est la valeur de b ?

Question 2) Dessiner l'arbre An

Matrice d'adjacence

Matrice booléenne A de taille $N \times N$

$A_{ij} = 1$ ssi l'arc (i, j) est dans le graphe

Taille : $O(N^2)$

Matrice de poids W pour un graphe pondéré

Avantages

- détection facile des boucles, de la symétrie ;
- test immédiat de l'existence d'un arc (i, j) ;
- énumération facile des prédécesseurs / successeurs.

Inconvénients

- Taille N^2 , même si graphe peu dense ($M \ll N^2$) ;
- désavantageux pour la complexité de certains algorithmes.

Listes d'adjacence

Codage par listes des successeurs

TSUC = liste des listes de successeurs

Exemple

$\text{Succ}(1) = \{2, 3, 4\}$, $\text{Succ}(2) = \{\}$, $\text{Succ}(3) = \{1, 2, 4\}$, $\text{Succ}(4) = \{4\}$

Avantages

Taille $N+M$ (intéressant si $M \ll N^2$) ;
avantageux pour la complexité de certains algorithmes ;
calcul facile de $d^+(i)$.

Inconvénients

pas de test immédiat de la présence d'un arc ;
pas d'accès direct aux prédécesseurs.

Principe

Chemin passant une et une seule fois par tous les sommets (graphe connexe)

Deux types classiques de parcours : Largeur et Longueur

Deux opérations supplémentaires

Marquer : sommet -> graphe Mise d'une marque sur un sommet donné

marque : sommet -> booléen Existence d'une marque sur un sommet donné

Algorithmes

Parcours en profondeur

Pour tout s de G faire si $\neg \text{marque}(s)$ alors Pprofondeur (G, s) fsi fpour
Pprofondeur :

Marquer(s)

Pour tout x de G tel que Arc(s, x) faire

si $\neg \text{marque}(x)$ alors Pprofondeur (G, x) fsi

fpour

Parcours en largeur

Pour tout s de G faire si $\neg \text{marque}(s)$ alors Plargeur (G, s) fsi fpour
Plargeur :

Marquer(s) f := Filevide Ajouter (f, s)

tantque $\neg \text{Estvide}(f)$ faire

p := tête(f) Retirer(f)

pour tout x de G tel que Arc(p, x) faire

si $\neg \text{marque}(x)$ alors Marquer(x) Ajouter (f, x) fsi

fpour

ftantque