Chapitre 2

Type de Données Abstraits (T.D.A.)

de la structure de données à l'encapsulation

Olivier Raynaud Université Blaise Pascal

Structure de données

Une **structure de données** est une manière d'organiser les données en mémoire pour en assurer une gestion efficace. Une structure de données implémente concrètement un type abstrait.

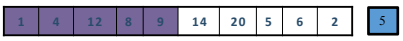
Wikipédia

- *Différentes classes*
  - ✓ **Structures finies** telles les constances, les variables, les enregistrements
  - ✓ **Structures indéxées** telles les tableaux,
  - ✓ **Structures récursives** telles les listes, les arbres ou les graphes

Structure de pile

Une pile est une structure de données fondée sur le principe que le dernier élément inséré sera le prochain élément à être traité. Une telle structure est aussi appelé LIFO (last-in, first out).

- **Opération usuelles :**
  - ✓ **« Empiler(x) »** : ajoute un élément sur la pile. Le terme anglais correspondant est *push*.
  - ✓ **« Dépiler »** : enlève un élément de la pile et le renvoie. Le terme anglais correspondant est *pop*.
  - ✓ **« La pile est-elle vide ? »** : renvoie vrai si la pile est vide, faux sinon



Inconvénient : il faut fixer à l'avance la taille maximale de la pile.

Structure de file

Une file, dite aussi **file** d'attente (en anglais queue), est une structure de données basée sur le principe du premier entré, premier sorti ou PEPS, (en anglais FIFO (« first in, firstout »). Autrement dit que les premiers éléments ajoutés à la **file** seront les premiers à en être retirés.

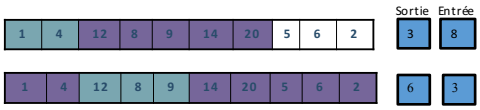
❑ Opération usuelles :

- ✓ **« Enfiler(x) »** : ajoute un élément dans la file. Le terme anglais correspondant est *enqueue*.
- ✓ **« Défiler »** : enlève un élément de la file et le renvoie. Le terme anglais correspondant est *dequeue*.
- ✓ **« La file est-elle vide ? »** : renvoie vrai si la pile est vide, faux sinon

Implémentation de la file

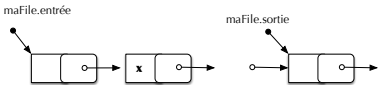
❑ Structure tabulaire

- ✓ **Un tableau** et deux entiers



❑ Structure récursive

- ✓ **Une structure composée** et deux pointeurs



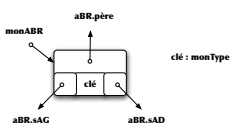
Structure d'arbre binaire de recherche

❑ *Structuration de la mémoire*

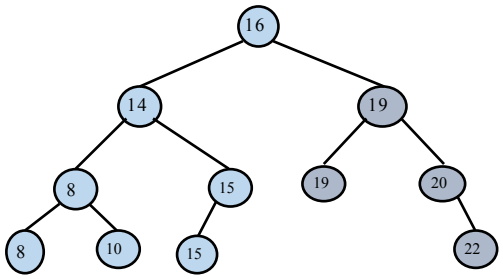
- ✓ **Un arbre** est une structure de stockage de l'information qui en permet la gestion de façon optimisée.
- ✓ **Opération usuelles :** l'insertion, la suppression, la recherche et l'accès aux éléments minimum et maximum.

Un arbre binaire est un A.B.R. si pour tout nœud s, les contenus des nœuds du sous-arbre de gauche sont inférieurs (≤) au contenu de s et les contenus du sous-arbre droit sont supérieurs (>) au contenu de s.

Accesseurs :
monABR.clé;,
monABR.sAG; monABR.sAD, monABR.père



Structure d'arbres binaires de recherche



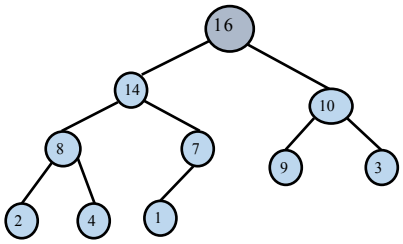
Structure de tas

- *Structuration de la mémoire*
  - ✓ **Un arbre** est une structure de stockage de l'information qui en permet la gestion de façon optimisée.
  - ✓ **Opération usuelles :** l'insertion, l'extraction et la recherche de l'éléments maximum (ou minimum).

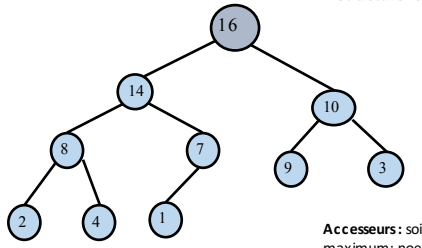
Une structure de tas est un arbre binaire presque complet tel que pour tout noeud n sauf la racine, la valeur affectée au père est supérieure ou égale à celles de ses fils.

Accesseurs : clé, fils Gauche, fils Droit et père

Structure de tas



Structure de tas



Accesseurs : soit i un noeud
maximum: noeud1; père(i) = i DIV 2;
fils Gauche(i) = 2.i; fils Droit (i) = 2.i+1;

monTas.taile=10
monTas.tab[]

1	2	3	4	5	6	7	8	9	10	
16	14	10	8	7	9	3	2	4	1	

Type de Données Abstrait

Un **type de données abstrait (T.D.A.)** est composé d'un ensemble d'objets, similaires dans la forme et dans le comportement, et d'un ensemble d'opérations sur ces objets.

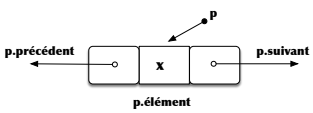
- *Choix d'implémentation*
  - ✓ **L'implémentation** d'un T.D.A. ne suis pas de schéma préétabli. Il dépend des objets manipulés et des opérations disponibles pour leur manipulation.
  - ✓ **Contraintes :** Utiliser un minimum d'espace mémoire; Exécuter un nombre minimal d'instructions pour réaliser une opération.

TDA - Ensemble dynamique

- *La gestion d'un ensemble d'objets*
  - ✓ **Un ensemble** rassemble un groupe d'objet similaires dans la forme et dans le comportement. La gestion d'un ensemble impose souvent de pouvoir lui adjoindre un nouvel élément ou d'en supprimer un par exemple.

On appelle **ensemble dynamique** *e* un ensemble fini d'éléments issus d'un ensemble discret (entiers, chaîne de caractères,...) et muni d'une relation d'ordre.

- ✓ **Opération usuelles :** l'insertion, la suppression, la recherche d'un élément, du minimum et du maximum, l'accès aux prédécesseurs et successeurs.



x, p.élément : monType
p, p.précédent, p.suivant : ensDynamique

TDA - Dictionnaire

On appelle **dictionnaire** un ensemble dynamique *d* dont on a restreint l'ensemble des opérations :

✓ **Opération usuelles :** l'insertion, la recherche et la suppression

Structure de données	Rechercher	Insérer	Supprimer
Tableau non ordonné	*O(n)*	*O(1)*	*O(1)*
Liste non ordonnée	*O(n)*	*O(1)*	*O(1)*
Tableau ordonné	*O(log n)*	*O(n)*	*O(n)*
Liste ordonnée	*O(n)*	*O(1)*	*O(1)*
Arbre de recherche	*O(h)*	*O(h)*	*O(h)*
Tas	*O(n)*	*O(h)*	*O(h)*

TDA - File de priorité

Une **file de priorité** est un type abstrait élémentaire permettant de gérer une liste d'éléments muni d'une valeur appelé *clé* suivant un ordre de priorité défini sur l'ensemble des clés.

❑ **Opération usuelles :**

✓ **« insérer(x, clé) »** : insère un élément x avec sa clé dans la file.

✓ **« maximum() »** : retourne l'élément de plus grande clé.

✓ **« extraireMax() »** : retourne et supprime l'élément de plus grande clé.

TDA - File de priorité

❑ *Implémentation et complexité*

✓ **Variable :** La clé est utilisée comme élément de sélection et d'ordonnancement

Structure de données	Insérer	Maximum	ExtraireMax
Tableau non ordonné	*O(1)*	*O(n)*	*O(n)*
Liste non ordonnée	*O(1)*	*O(n)*	*O(n)*
Tableau ordonné	*O(n)*	*O(1)*	*O(n)*
Liste ordonnée	*O(n)*	*O(1)*	*O(1)*
Tas	*?*	*?*	*?*

TDA – Famille d'ensembles

❑ **Famille d'ensemble on usuelles :**

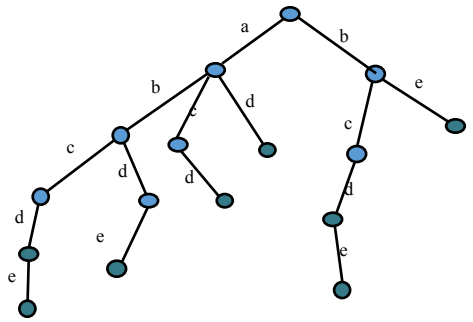
✓ Soit **X** un ensemble muni d'une relation d'ordre **<x**, on appelle **collection** *(ou famille)* un ensemble F de sous-ensembles de X.

Le T.D.A. **famille d'ensembles** permet la gestion et la manipulation d'une famille d'ensembles issue d'un ensemble X.

❑ **Opération usuelles :**

✓ **« insérer(e) »** : insère l'ensemble e dans la famille.
✓ **« appartient?(e) »** : retourne vrai si l'ensemble **e** est dans la famille.
✓ **« supprimer(e) »** : supprime l'ensemble **e**.

TDA – Famille d'ensembles



Olivier Raynaud, Université Blaise Pascal, Clermont-Ferrand

TDA – Famille d'ensembles

❑ **Implémentation abstraite - Modélisation:**

✓ L'ensemble **X** est totalement ordonné, on appelle **<x** cette relation d'ordre.

Soit F une famille de sous-ensembles de X,
nous associons à F un arbre lexicographique T(F) unique tel que :

- chaque arête de l'arbre est étiqueté par un élément de X;
- à chaque nœud notifié de l'arbre correspond un mot de F;
- à chaque mot de F correspond un chemin unique dans l'arbre tel que ce mot corresponde à la concaténation des étiquettes de ce chemin;
- l'ordre des arêtes d'un chemin coïncident avec l'ordre $<_x$;
- l'ordre des arêtes sortant d'un nœud coïncident avec l'ordre $<_x$.

TDA – Famille d'ensembles

❑ **Implémentation concrète – Structure en Java**

✓ **Remarque :** une représentation d'une collection par un arbrelexicographique correspond à un mapping !

Java Key Mapping :
new() operator : crée un objet de type map et retourne un mapping vide;

get(e) operator : retourne la valeur associée à la clé *e* si cette clé existe, nil dans le cas contraire;

put(e, value) operator : insère la clé *e* dans le map et lui associe la valeur *value*.

TDA – Famille d'ensembles

	CPU times (milliseconds)								
	ADT L			ADT M			ADT T		
Full(n)	put	get	remove	put	get	remove	put	get	remove
12	103	71	56	142	27	30	33	9	16
13	338	45	47	281	10	25	69	9	24
14	384	35	109	351	11	21	116	9	19
15	474	34	38	568	11	21	255	11	29
16	755	28	38	868	11	23	535	11	29
17	1771	38	47	1480	13	27	1132	12	40
18	5251	66	41	5692	41	65	3191	28	40
19	7240	29	49	7407	13	30	6083	12	19
20	15212	40	41	16310	14	32	12965	11	20

Table . Experimental evaluation on Full examples using ADT MappingList, ADT MappingMap and ADT MappingTable

TDA – Gestion de partition

❑ **Définition**

✓ **Rappel :** une partition **P** d'un ensemble **E,** est un ensemble de parties non vides de ***E,*** deux à deux disjointes et dont la réunion est égale à E

Le T.D.A. **Gestion de partition** permet la gestion d'une partition *dynamique* issue d'un ensemble **E** initial. Plus précisément, la partition évolue de l'ensemble des singletons vers une partition composée d'une seule partie E.

❑ **Opération usuelles :** e est un élément de E

✓ **« trouverClasse(e)» :** retourne la classe de **e** dans la partition.

✓ **« union(C_1, C_2) » :** fusionne les deux classes C_1 et C_2 dans la partition

TDA – Gestion de partition

- Implémentation
  - ✓ **Tableau**

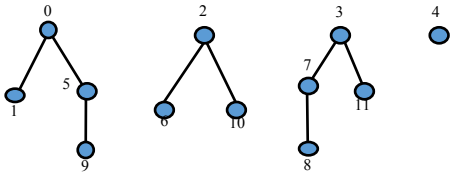
1	1	2	3	4	1	2	3	3	1	2	3
0	1	2	3	4	5	6	7	8	9	10	11

Partition : { {0,1,5,9}, {2,6,10}, {3,7,8,11}, {4} }

Question : Quelle est la complexité des opérations de fusion et de recherche?

TDA – Gestion de partition

- Implémentation
  - ✓ **Collection d'arbres**



Partition : { {0,1,5,9}, {2,6,10}, {3,7,8,11}, {4} }

Père :

0	0	2	3	4	0	2	3	7	5	2	3
0	1	2	3	4	5	6	7	8	9	10	11

Question : Quelle est la complexité des opérations de fusion et de recherche?

Pour résumer

- *Quelques points essentiels de l'exposé*
  - ✓ **Structure de données :** Nous avons rappelé les principes abstraits et d'implémentation de plusieurs structures de données classiques telles que les *Piles*, les *Files*, les *Arbre Binaires de Recherche* et les *Tas*.
  - ✓ **Type de Données Abstraits :** Nous avons défini les T.D.A. comme la donnée d'un ensemble d'objets cohérent muni d'une liste d'opérations sur ces objets. Nous avons identifié les T.D.A. *Ensemble Dynamique*, *Dictionnaire*, *File de Priorité*, *Famille d'Ensembles* et *Gestion de Partition*.
  - ✓ **Implémentation :** Nous avons montréet donné plusieurs implémentations possibles pour ces T.D.A. Chacunede ces implémentations repose sur des structures de données classiques telles que les listes, les tableaux, les arbre, les tas, ou les arbres binaires de recherche.