

# Troisième cours

## Structures complexes



4F1 21B2C809 8833B0C 2957E  
CAA CB3EE8EF DF038E A142J  
A4D 04143B75 4F57E83 535C0  
ED9 B57C659F EE07 FA49  
5DB 7D 9A 6DD29 454E  
.4D 410 9 54E072 5A14  
52 534 860929 D8E  
FC 0F1 C A60B99 4420  
78 E08EDA 457266E E71  
81 B5928D82 6C9C0575 286  
78 CF26B3CA FD6C4411 BE7  
AB D41F4256 0400312E 300

# Un petit jeu pour les enfants.

## Mes bateaux

Nombre de coups nécessaires :

 9058	 7169	 3214	 5891	 4917	 2767	 4715	 674	 8088	 1790	 8949	 13	 3014
<b>A</b>	<b>B</b>	<b>C</b>	<b>D</b>	<b>E</b>	<b>F</b>	<b>G</b>	<b>H</b>	<b>I</b>	<b>J</b>	<b>K</b>	<b>L</b>	<b>M</b>
 8311	 7621	 3542	 9264	 450	 8562	 4191	 4932	 9462	 8423	 5063	 6221	 2244
<b>N</b>	<b>O</b>	<b>P</b>	<b>Q</b>	<b>R</b>	<b>S</b>	<b>T</b>	<b>U</b>	<b>V</b>	<b>W</b>	<b>X</b>	<b>Y</b>	<b>Z</b>

## Tes bateaux

Nombre de coups nécessaires :

												
<b>A</b>	<b>B</b>	<b>C</b>	<b>D</b>	<b>E</b>	<b>F</b>	<b>G</b>	<b>H</b>	<b>I</b>	<b>J</b>	<b>K</b>	<b>L</b>	<b>M</b>
												
<b>N</b>	<b>O</b>	<b>P</b>	<b>Q</b>	<b>R</b>	<b>S</b>	<b>T</b>	<b>U</b>	<b>V</b>	<b>W</b>	<b>X</b>	<b>Y</b>	<b>Z</b>

# Une autre partie ?

**Mes bateaux** Nombre de coups nécessaires :

 163	 445	 622	 1410	 1704	 2169	 2680	 2713	 2734	 3972	 4208	 4871	 5031
<b>A</b>	<b>B</b>	<b>C</b>	<b>D</b>	<b>E</b>	<b>F</b>	<b>G</b>	<b>H</b>	<b>I</b>	<b>J</b>	<b>K</b>	<b>L</b>	<b>M</b>
 5283	 5704	 6025	 6801	 7440	 7542	 7956	 8094	 8672	 9137	 9224	 9508	 9663
<b>N</b>	<b>O</b>	<b>P</b>	<b>Q</b>	<b>R</b>	<b>S</b>	<b>T</b>	<b>U</b>	<b>V</b>	<b>W</b>	<b>X</b>	<b>Y</b>	<b>Z</b>

**Tes bateaux** Nombre de coups nécessaires :

												
<b>A</b>	<b>B</b>	<b>C</b>	<b>D</b>	<b>E</b>	<b>F</b>	<b>G</b>	<b>H</b>	<b>I</b>	<b>J</b>	<b>K</b>	<b>L</b>	<b>M</b>
												
<b>N</b>	<b>O</b>	<b>P</b>	<b>Q</b>	<b>R</b>	<b>S</b>	<b>T</b>	<b>U</b>	<b>V</b>	<b>W</b>	<b>X</b>	<b>Y</b>	<b>Z</b>

## Complexité d'un algorithme

Définition :

**Complexité en temps** : Quantité de temps nécessaire à l'algorithme pour s'exécuter

**Complexité en espace** : Quantité de mémoire nécessaire à l'algorithme pour s'exécuter.

# Complexité d'un algorithme

## Définition :

**Complexité en temps** : Quantité de temps nécessaire à l'algorithme pour s'exécuter

**Complexité en espace** : Quantité de mémoire nécessaire à l'algorithme pour s'exécuter.

- ▶ fortement dépendant de la machine, des éléments utilisés
- ▶ plutôt en terme d'ordre de grandeur
- ▶ en fonction de la taille de l'entrée

# Complexité d'un algorithme

## Définition :

**Complexité en temps** : Quantité de temps nécessaire à l'algorithme pour s'exécuter

**Complexité en espace** : Quantité de mémoire nécessaire à l'algorithme pour s'exécuter.

- ▶ fortement dépendant de la machine, des éléments utilisés
- ▶ plutôt en terme d'ordre de grandeur
- ▶ en fonction de la taille de l'entrée

**Notation de Landau** : complexité en  $O(f(n))$

→ il existe une constante  $C$  telle que la complexité  $\leq C \times f(n)$ .

## Retour sur la bataille navale :

Mes bateaux													Nombre de coups nécessaires :	
 9058	 7169	 3214	 5891	 4917	 2767	 4715	 674	 8088	 1790	 8949	 13	 3014		
<b>A</b>	<b>B</b>	<b>C</b>	<b>D</b>	<b>E</b>	<b>F</b>	<b>G</b>	<b>H</b>	<b>I</b>	<b>J</b>	<b>K</b>	<b>L</b>	<b>M</b>		
 8311	 7621	 3542	 9264	 450	 8562	 4191	 4932	 9462	 8423	 5063	 6221	 2244		
<b>N</b>	<b>O</b>	<b>P</b>	<b>Q</b>	<b>R</b>	<b>S</b>	<b>T</b>	<b>U</b>	<b>V</b>	<b>W</b>	<b>X</b>	<b>Y</b>	<b>Z</b>		

### Structure non organisée

- ▶  $n$  appels dans le pire cas
- ▶  $\frac{n}{2}$  en moyenne
- ▶ Recherche en  $O(n)$

## Dichotomie

Mes bateaux													Nombre de coups nécessaires :
													
163	445	622	1410	1704	2169	2680	2713	2734	3972	4208	4871	5031	
<b>A</b>	<b>B</b>	<b>C</b>	<b>D</b>	<b>E</b>	<b>F</b>	<b>G</b>	<b>H</b>	<b>I</b>	<b>J</b>	<b>K</b>	<b>L</b>	<b>M</b>	
													
5283	5704	6025	6801	7440	7542	7956	8094	8672	9137	9224	9508	9663	
<b>N</b>	<b>O</b>	<b>P</b>	<b>Q</b>	<b>R</b>	<b>S</b>	<b>T</b>	<b>U</b>	<b>V</b>	<b>W</b>	<b>X</b>	<b>Y</b>	<b>Z</b>	

- ▶ Structure indexée.
- ▶ Éléments dans l'ordre.

### Structure triée

- ▶ Recherche en  $O(\log_2(n))$  (dans le pire cas et en moyenne).

# Encore une partie ?

Mes bateaux										Nombre de coups nécessaires :									
0		1		2		3		4		5		6		7		8		9	
A	9047	C	3080			E	5125	H	8051	L	7116	O	6000	R	9891			W	1062
B	1829	D	9994			F	1480	I	1481	M	8944	P	7432	S	1989	V	4392	X	2106
						G	8212	J	4712	N	4128	Q	4110	T	2050			Y	5842
								K	6422					U	8199			Z	7057

Tes bateaux										Nombre de coups nécessaires :									
0		1		2		3		4		5		6		7		8		9	
A		E		H				L				O		R		V			
B		F		I		K		M				P		S		W		Y	
C		G		J				N				Q		T		X		Z	
D														U					

CC BY-NC-SA Computer Science Unplugged (csunplugged.org) 2005-2014

## Recherche par clé

- ▶ Pour chaque valeur, une clé est calculée (appelé *hash*).
- ▶ Pour chaque clé, l'ensemble des éléments qui correspondent (par exemple liste chaînée).

### Table de hashage

- ▶ Fort rôle de la taille de l'espace des clés disons  $k$  clés.
- ▶ Recherche en  $O(\frac{n}{k})$  en moyenne.

# Fonction de Hashage

## Attributs recherchés

- ▶ clé décorélée de la valeur initiale (ensemble de valeurs homogènes)
- ▶ Peu de collisions (éléments ayant la même image)

## Usages

- ▶ HashSet
- ▶ HashMap, dictionnaire (hash de la clé).
- ▶ Cryptographie (Signature, stockage de mot de passe)

## Comparaison des structures de données

**Tableau** : pré-affecté en mémoire,  $n$  cases statiques (e.g. ArrayList)

**Liste chaînée** : structure chaînée, dynamique (e.g. LinkedList)

**Table de hachage** : Tableau de listes chaînées (e.g. HashSet)

# Comparaison des structures de données

## Complexité des opérations

Opération	Tableau	Liste	Tableau trié	Hashset
Accès par position				
Recherche				
Insertion/suppression				

# Comparaison des structures de données

## Complexité des opérations

Opération	Tableau	Liste	Tableau trié	Hashset
Accès par position	$O(1)$	$O(n)$	$O(1)$	N/A
Recherche	$O(n)$	$O(n)$	$O(\log(n))$	$O(n/k)$
Insertion/suppression	$O(n)$	$O(1)$	$O(n+\log(n))$	$O(1)$

En pratique, des techniques d'accélération.

# Complexité en temps

Exemples et vocabulaire :

Problème	Complexité		$n = 50$
Accéder à une case d'un tableau	$O(1)$	[constant]	10ns
Chercher dans un tableau trié	$O(\log(n))$	[logarithmique]	20ns
Chercher dans un tableau	$O(n)$	[linéaire]	500ns
Multiplier des matrices	$O(n^3)$	[cubique]	1,25ms
Résoudre un Sudoku à $n$ cases	$O(9^n)$	[exponentiel]	130 jours

Pour un problème dont l'entrée est de taille  $n$

**Algorithme polynomial** : Complexité en  $O(n^k)$ , assez rapide

**Algorithme Exponentiel** : Complexité en  $O(k^n)$ , très lent.

# NP-complétude

## Culturel

### Problème NP-complet

- ▶ Pas d'algorithme polynomial connu
- ▶ Si on en trouve un, on les résout tous.

### Exemples de problèmes NP-complets

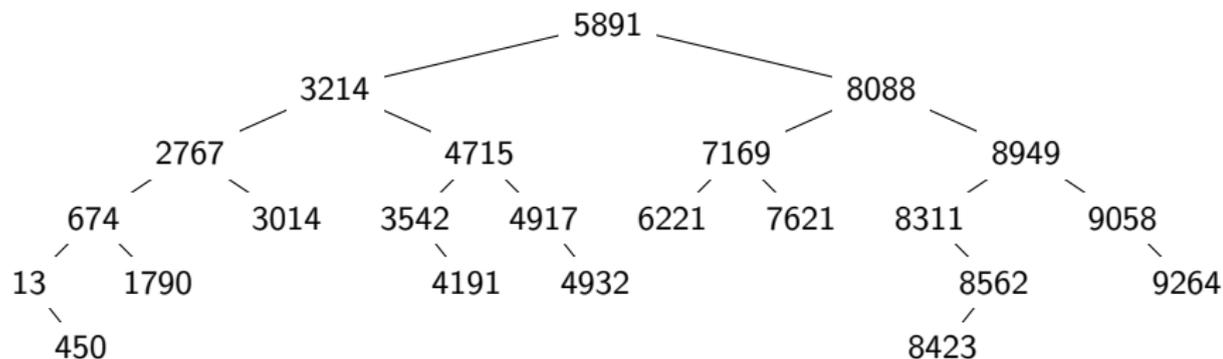
**Voyageur de commerce** plus court chemin qui visite  $n$  points fixés.

**Emploi du temps** qui satisfait les contraintes de salles/prof/élèves

**Factoriser un grand nombre** (permet de casser le **chiffrement RSA**)

**Plein de casse-têtes** (Sokoban, démineur, mario...)

# Arbre binaire de recherche



# Arbre binaire de recherche

## Principes

- ▶ Structure d'arbre

## Usages

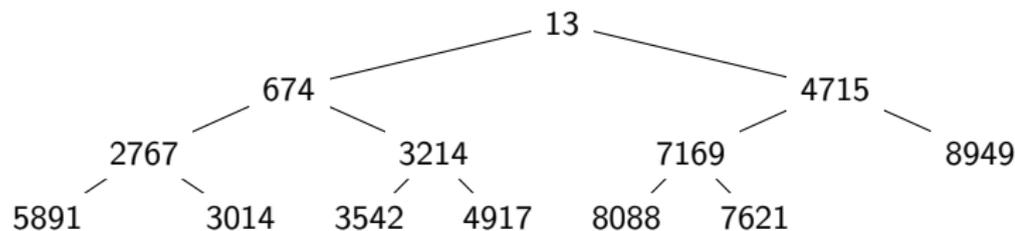
- ▶ insertion, recherche au même prix
- ▶ Complexité en moyenne :  $O(\log(n))$
- ▶ Dans le pire cas :  $O(n)$ .
- ▶ Suppression délicate avec rotation

Extension de la rotation pour équilibrer l'arbre.

## Les tas

### Définition :

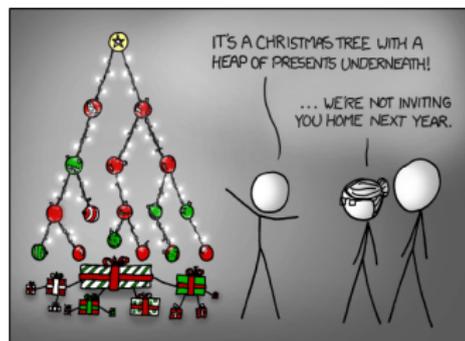
- ▶ Une structure semi-triée rangée dans un tableau
- ▶ Règle du jeu : chaque nœud est plus petit que ses enfants.
- ▶ en Java, implémentée dans les `PriorityQueue`.



# Synthèse

## À retenir

- ▶ Expression de la complexité algorithmique.
- ▶ Chacune des structures de données classiques.
- ▶ Savoir faire un choix éclairé!
- ▶ D'autres structures arborescentes existent, plus riches.



<https://xkcd.com/835/>