Institut Universitaire de Technologie
Aix*Marseille Université

M3103
Algorithmique avancée

Chapitre 01 - Les tables de hachage

[brett.desbenoit | laurent.carmignac]@univ-amu.fr

2018-2019

Problématique

Comment manipuler efficacement un ensemble non ordonné ?

Considérons, par exemple, le cas d'un annuaire. On souhaite stocker et manipuler un ensemble de personnes (nom, prénom, numéro de téléphone).

Quelle structure de données utiliser ?
Tableau de structures ?
Listes chainées ?

Quelle est son efficacité en lecture / écriture ?
Est-il possible d'avoir un accès direct aux éléments en lecture ?
Quel coût représente l'accès ? La recherche ? la suppression ?

Comment améliorer les performances en terme de stockage ?
Peut-on utiliser l'allocation dynamique ?
Quel est l'impact des données éparses sur la mémoire ?

Définitions

[brett.desbenoit|laurent.carmignac]@univ-amu.fr

Définitions

Table de symboles

Une table de symboles permet de stocker efficacement des paires $< clés/valeurs >$. Par exemple, si l'on considère un répertoire téléphonique, la clé est le nom de la personne et la valeur le numéro de téléphone.

Ce type de table permet par exemple de créer des tableaux associatifs ; mais pas que !

Tables associatives

Table de symbole dans laquelle les clés sont uniques (Annuaire de personnes avec pour clé le numéro de sécurité social, pseudo de connexion, etc.).

Dictionnaires

Les clés des dictionnaires quant à elles ne sont pas uniques (Répertoire téléphonique, dictionnaire de synonymes, etc.).

Définitions

Table de hachage

Une table de hachage est une **implantation de l'interface d'une table de symboles** pour un type de données particulier.

Il existe en faite différentes implantations possible d'une table de symboles :

- Les tableaux ;
- les listes ;
- Les arbre ;
- Et les tables de hachage.

La table de hachage est préférée dans le cas de données non ordonnées. D'autres implantations sont possibles comme par exemple une implantation à base d'arbre. Cette dernière offre de meilleurs performances dans le cas de données ordonnées.

Les tables de hachage

[brett.desbenoit|laurent.carmignac]@univ-amu.fr

Les tables de hachage

Type abstrait

TA: TableHachage

Utilise: $Booléen, Entier, Clé, Valeur$

Opérations:

insérer: $TableHachage \times Clé \times Valeur \rightarrow TableHachage$

rechercher: $TableHachage \times Clé \rightarrow Valeur$

supprimer: $TableHachage \times Clé \rightarrow TableHachage$

contient: $TableHachage \times Clé \rightarrow Booléen$

taille:$TableHachage \rightarrow Entier$

estVide: $TableHachage \rightarrow Booléen$

Pré-conditions:

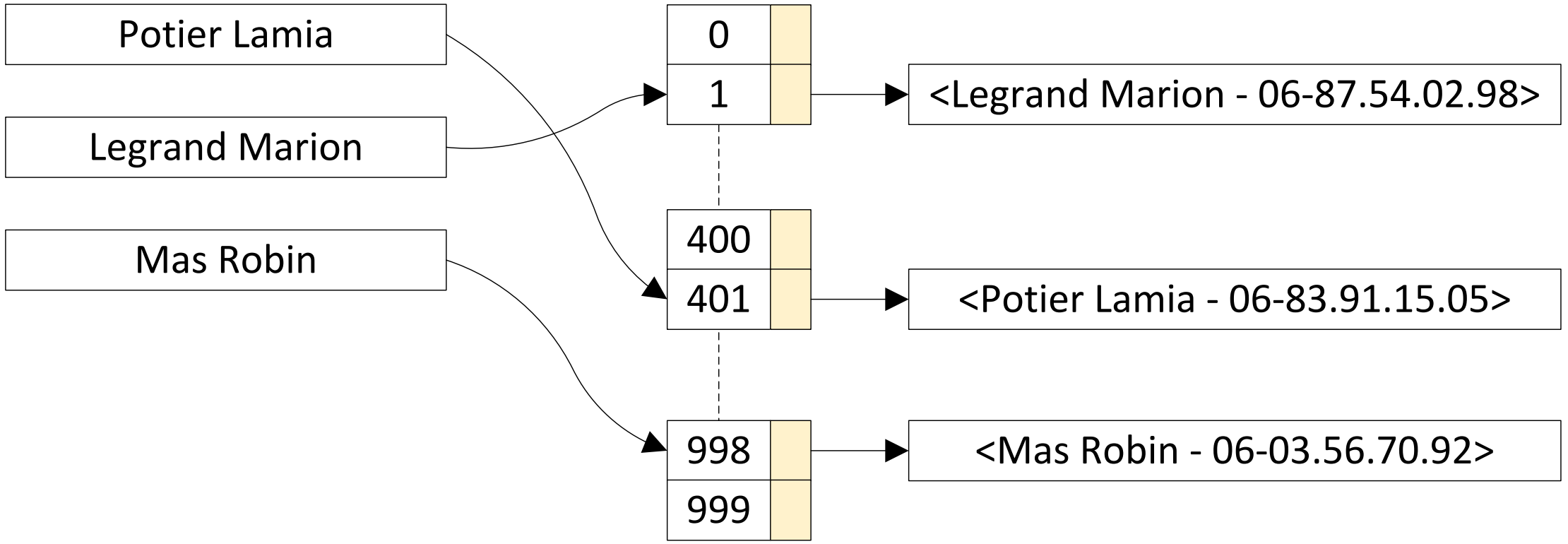
…

Propriétés

1. On **accède** à chaque **élément** de la table **via sa clé** ;
2. L'accès à un élément se fait en **transformant** la **clé** en **une valeur de hachage** (hash code) par l'intermédiaire d'une **fonction de hachage** ;
3. La **valeur de hachage** est un **nombre entie**r qui permet la localisation des éléments dans le tableau (par son indice par exemple). Les cases de ce tableau sont appelées **alvéole**.

Les tables de hachage

Exemple



Remarque : une table de hachage n'est pas ordonnée.

Les tables de hachage

Complexité

Table de la complexité dans le meilleur des cas des opérations de lecture, écriture et suppression pour les différentes implantation d'une table de symboles

	Tableau trié	Tableau non trié	Liste triée	Liste non triée	Arbre	Table de hachage
Ajout	$O(n)$	$O(1)$	$O(n)$	$O(1)$	$O(\log(n))$	$O(1)$
Recherche	$O(\log(n))$	$O(n)$	$O(n)$	$O(n)$	$O(\log(n))$	$O(1)$
Suppression	$O(\log(n))$	$O(n)$	$O(n)$	$O(n)$	$O(\log(n))$	$O(1)$

Les tables de hachage permettent un accès en $O(1)$ dans le meilleur des cas et en $O(n)$ dans le pire cas.

La fonctions de hachage

[brett.desbenoit|laurent.carmignac]@univ-amu.fr

La fonction de hachage

Propriétés

Une fonction de hachage h doit respecter les deux propriétés suivantes :

- Si deux clés sont identiques, leur hash code est identique :
  pour deux clés k_1 et k_2, si $k_1 = k_2$ alors $h(k_1) = h(k_2)$
- les hash codes doivent être tous unique :
  pour deux clés k_1 et k_2, si $k_1 \neq k_2$ alors $h(k_1) \neq h(k_2)$

Il faut choisir soigneusement sa fonction de hachage !

Fonction de compression

On peut utiliser des fonctions de compression pour construire des fonctions de hachage. Une fonction de compression permet de passer d'un ensemble de valeurs quelconques à un ensemble de taille fixe. Ainsi, si l'on considère K un ensemble de valeurs et m un entier naturel, une fonction de hachage h_m construite par une fonction de compression est une fonction définie de K dans $\{0, \ldots, m-1\}$.

Quelle fonction de compression connaissez-vous déjà ?

La fonction de hachage

Méthode par division

La méthode par division consiste simplement à borner l'ensemble en utilisant la fonction modulo :

$\forall k \in \mathbb{N}, h_m(k) = k \bmod m$ avec m pas trop proche d'une puissance de 2.

Le problème ici est que cela peut produire beaucoup de collisions. Pour éviter ça, on peut utiliser le calcul suivant :

$\forall k \in \mathbb{N}, h_m(k) = (k \bmod p) \bmod m$ avec p un nombre premier tel que $p > m$.

Méthode par multiplication

On multiplie la clé par une constante $c \in [0, 1[$, on prend la partie entière, on multiplie par m et on prend la valeur inférieur la plus proche.

$\forall k \in \mathbb{N}, h_{m,c}(k) = \lfloor m \times \lfloor kc \rfloor \rfloor$ avec $m = 2^p$ et $c = \frac{\sqrt{5}-1}{2}$

Fonctions de hachage

Hachage polynomial

Une fonction de hachage polynomiale est une fonction qui est basée sur la somme de la valeur de chaque caractère. Si l'on considère la fonction asc qui a un caractère associe son code ASCII codé sur 7 bits et $c_0 \dots c_p$ une suite de caractères alors :

$$h = \sum_{i=0}^{p} asc(c_i) \times 128$$

Le problème de cette fonction de hachage est que les anagrammes (marie, aimer, maire, etc.) auront nécessairement le même hash code. Pour éviter ce problème, on peut porter la constante à une puissance quelconques :

$$h = \sum_{i=0}^{p} asc(c_i) \times 128^i$$

Hachage cyclique

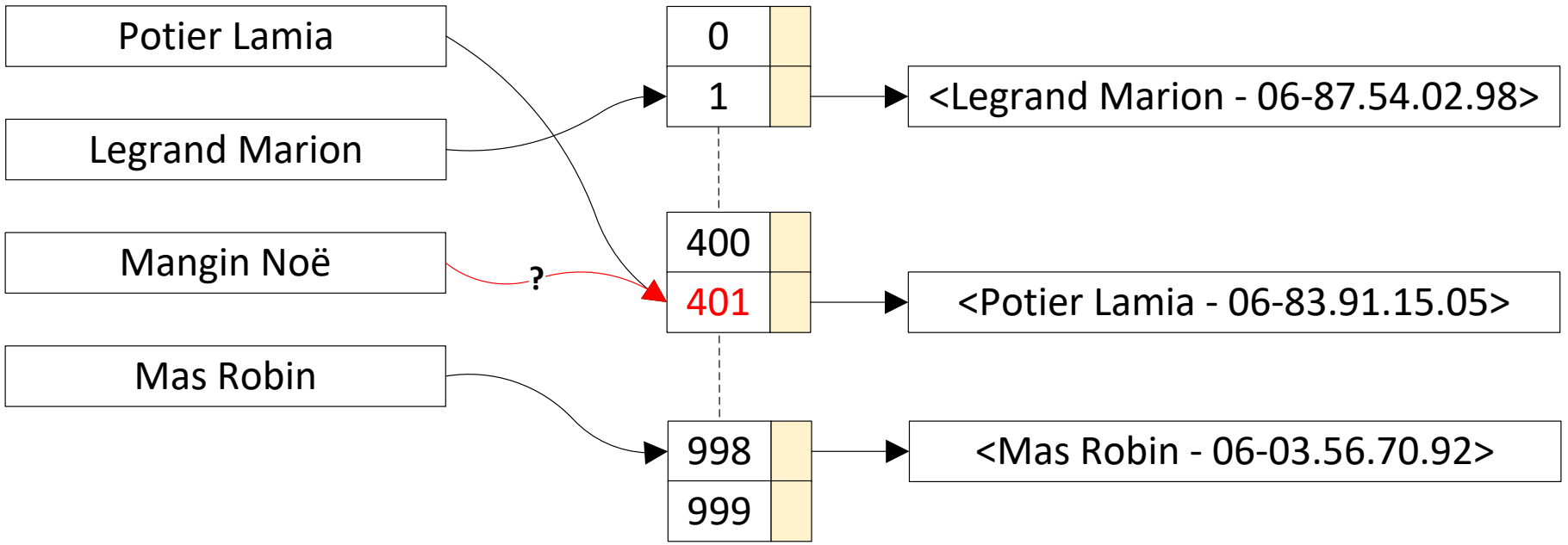
Le hachage cyclique consiste à opérer par décalage de bits : on prend les n premiers bits de poids fort et on les mets à la place des n derniers bits de poids faible puis on ajoute la valeur numérique du caractère.

Le problème des collisions

[brett.desbenoit|laurent.carmignac]@univ-amu.fr

Les problèmes de collisions

Lorsque deux clés k_1 et k_2 vérifient $k_1 \neq k_2$ et $h_m(k_1) = h_m(k_2)$, on dit qu'il y a **collision**.



Les problèmes de collisions

La probabilité d'avoir une collision n'est pas négligeable. Si l'on fait l'hypothèse que l'on a une répartition uniforme des données dans la table de hachage, alors, ce problème est similaire **au problème des anniversaires** :

> *“ Dans un groupe de personnes, quelle est la probabilité que deux soient nées le même jour ?”*

Les problèmes de collisions

Problème des anniversaires

Pour calculer cette probabilité P, on estime la probabilité P' que les personnes soient toutes nées des jours différents :

$$P'(n) = \frac{365 \times 364 \times 363 \times \cdots \times (365 - n - 1)}{365^n} = \frac{1}{365^n} \times \prod_{i=0}^{n} (365 - i + 1)$$

On a alors $P(n) = 1 - P'(n)$ soit :

$$P(n) = 1 - \frac{1}{365^n} \times \prod_{i=0}^{n} (365 - i + 1)$$

Exemple pour quelques valeurs de n :

n	5	10	15	20	25	30	40	50	60	80	100
$P(n)$	2,71%	11,69%	25,99%	41,14%	56,87%	70,63%	89,12%	97,04%	99,41%	99,99%	99,99997%

Les problèmes de collisions

Retour aux tables de hachage

Si l'on considère une table de hachage de taille m et contenant déjà n éléments, alors la probabilité qu'il y ai une collision est de l'ordre de :

$$P(n) = 1 - \frac{1}{m^n} \times \prod_{i=0}^{n} (m - i + 1)$$

La gestion des collisions

Adressage ouvert

[brett.desbenoit|laurent.carmignac]@univ-amu.fr

2018-2019

Adressage ouvert

Principe

L'adressage ouvert consiste dans le cas d'une collision à stocker les éléments dans d'autres alvéoles de la table :

- Toutes les valeurs sont conservées dans la table ;
- Si une alvéole est déjà occupée, on en cherche une autre ;
- La position de ces alvéoles est déterminée par une méthode de *sondage*.

Pour la recherche, si la case obtenue par hachage direct ne permet pas d'obtenir le bon élément, une recherche sur les cases obtenues par la méthode de sondage est effectuée jusqu'à trouver l'élément. Si l'on arrive au bout du processus sans trouver l'élément c'est qu'aucun élément de ce type n'appartient à la table.

Adressage ouvert

Méthodes de sondage

Parmi les méthodes de sondages utilisée avec l'adressage ouvert on peut citer :

- Le sondage linéaire ;
- le sondage quadratique ;
- le double hachage.

Sondage linéaire

Principe

En cas de collision, le sondage linéaire consiste à parcours les alvéoles suivantes jusqu'à ce que l'on trouve une alvéole vide. Pour la recherche, le principe est similaire : on parcours toutes les alvéoles à partir de l'indice donné par le hash code. Si on arrive sur une alvéole vide c'est que l'élément n'est pas dans la table.

Un exemple de fonction de sondage linéaire est : $s(k,i) = (h(k) + i\ ) \bmod m$ avec i le nombre de tentatives et m la taille du tableau.

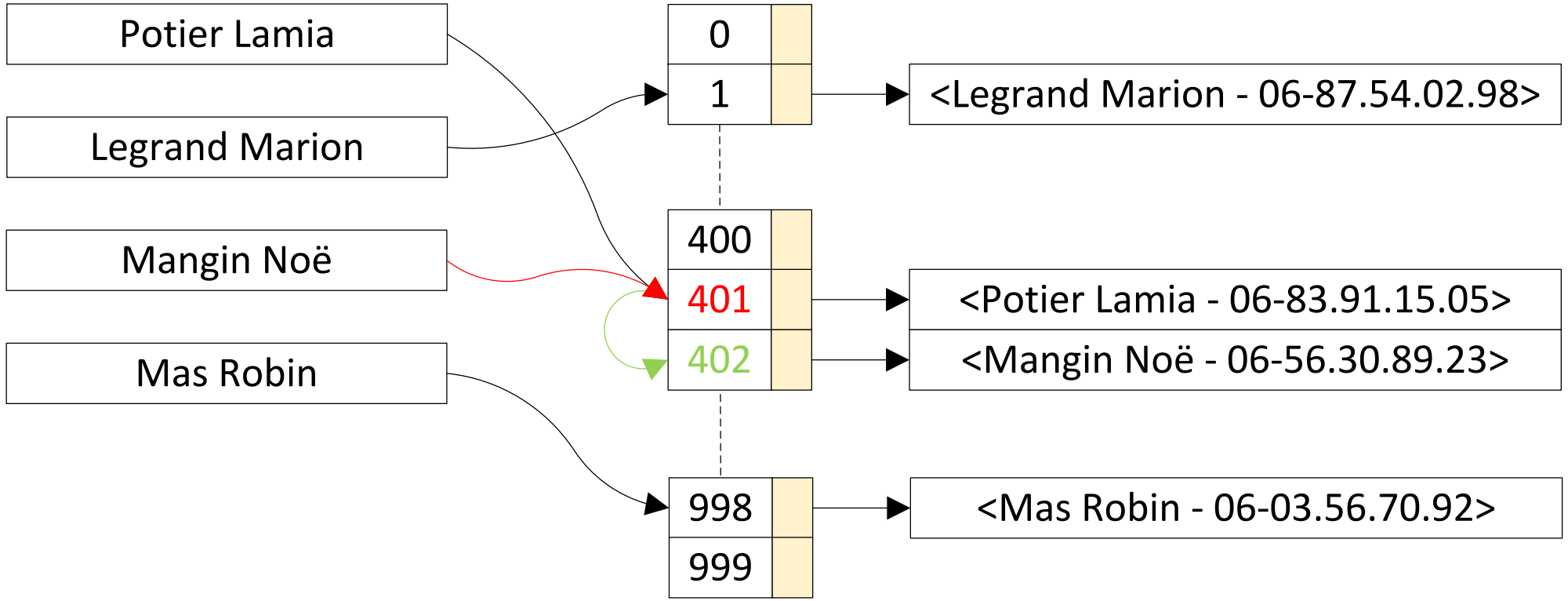
Cela va conduire à parcourir le tableau en commençant à l'indice $[h(k)]$, puis $[h(k)+1]$, $[h(k)+2]$,…

Remarque :

Si au bout de m sondages, aucune place disponible n'a été détectée, la table est dite saturée.

Sondage linéaire

Exemple



Sondage linéaire

Avantages :

- Cette méthode de sondage présente l'avantage d'être très simple à calculer ;
- Elle fonctionne très bien quand la table est peu remplie.

Inconvénients :

- Deux clés ayant le même hash code auront la même suite d'adresses ;
- Cela va conduire à la formation de clusters ce qui est a éviter.

Sondage quadratique

Principe

Afin d'éviter la formation de *clusters*, on peut utiliser une fonction de sondage quadratique. Contrairement au sondage linéaire où l'intervalle entre les alvéoles est fixe, ici l'intervalle augmente de manière linéaire et donc les indices de manière quadratique. Cela va conduire à mieux répartir les éléments au sein de la table.

$$s(k,i) = (h(k) + a \times i + b \times i^2) \bmod m$$

avec i le nombre de tentatives, a et b des constantes non nulles, et m la taille du tableau.

Avec $a = 1$ et $b = 1$, cela va conduire à parcourir le tableau en commençant à l'indice $[h(k)]$, puis $[h(k) + 2]$, $[h(k) + 6]$, $[h(k) + 12]$, …

Les données seront mieux réparties. Cependant, les clés de même hash code utiliseront le même parcours dans la table. Cela conduit à la formation de *clusters* secondaires. Néanmoins, ces clusters sont moins important que dans le cas du sondage linéaire.

Sondage quadratique

Avantages

- plus efficace que le sondage linéaire ;
- moins d'effet de formation de groupements.

Inconvénients

- Il est difficile de choisir les valeurs de a, b et m pour s'assurer de parcourir toute la table ;
- comme pour le sondage linéaire, deux clés ayant le même hash code auront la même suite d'adresses.

Double hachage

Principe

La méthode par double hachage consiste à utiliser deux fonctions de hachage pour déterminer l'indice d'un élément :

$$s(k,i) = \left(h_{1(k)} + i \times h_{2(k)}\right) \bmod m$$

L'introduction de $i \times h_2(k)$ va permettre d'éviter la création de grappe en faisant varier la clé en fonction du nombre de tentatives.

Exemple

Avec m un nombre premier et m' tel que m' soit légèrement inférieur à m.

$$\begin{aligned} m &= 2^p \\ h_1(k) &= k \bmod m \\ h_2(k) &= 1 + k \bmod m' \end{aligned}$$

Double hachage

Avantages

- La suite d'adresse ne sera pas forcement linéaire ;
- meilleur répartition des éléments dans la table.

Inconvénients

Il faut que $h_2(k)$ soit premier avec m sinon on ne parcourra pas toutes les alvéoles.

Sondage linéaire, quadratique et double hachage

Bilan

- Le sondage linéaire est facile à calculer mais génère rapidement des clusters.
- Le double hachage est plus complexe mais permet de réduire presque complètement les problèmes de clustering.
- Le sondage quadratique se situe entre le linéaire et le double hachage au niveau des performances mais il n'est pas évident de garantir que l'on parcours toute la table.

Adressage ouvert

Bilan

- l'insertion et la recherche d'un élément est facile.
- la suppression d'un élément n'est pas trivial et nécessite de passer par un marquage ou un réarrangement de la table.
- Dans le meilleur des cas, la recherche est en $O(1)$, et dans le pire des cas en $O(n)$.

La gestion des collisions

chaînage linéaire

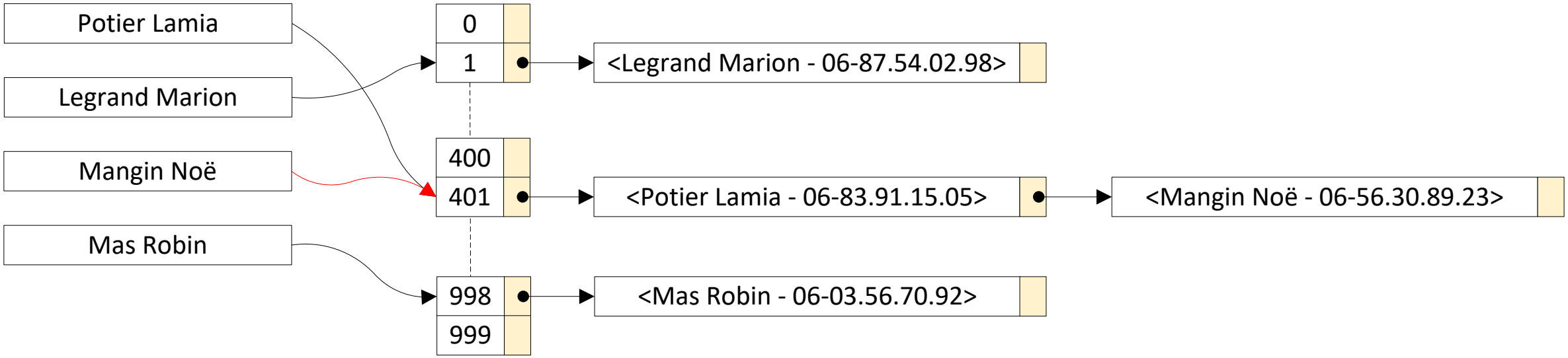
[brett.desbenoit|laurent.carmignac]@univ-amu.fr

2018-2019

Le chaînage linéaire

Principe

- Chaque alvéole de la table devient une liste ;
- Les valeurs sont stockées à l'extérieur de la table qui est généralement une liste chaînée de pointeurs ;
- La table a la capacité de grandir ;
- Si une alvéole est déjà occupée, on ajoute dans la même alvéole le nouveau couple $< clé, valeur >$.



Le chaînage linéaire

Bilan

- Permet de stocker autant d'éléments que l'on veut.
- La performance dépend à la fois de la fonction de hachage et de la taille de la table.
- L'insertion, la recherche et la suppression sont faciles : gestion de listes chaînées.
- Dans le pire des cas, recherche en $O(n)$.

Le facteur de charge

[brett.desbenoit|laurent.carmignac]@univ-amu.fr

Facteur de charge

Problématique

Quelle que soit la méthode de hachage utilisée, plus il y a de clés, plus il y aura de risque de collision et plus les temps de recherche seront grands :

- avec un sondage (linéaire ou quadratique), les clusters seront de plus en plus importants ;
- avec un double hachage, le nombre d'itérations augmentent ;
- avec un chaînage linéaire, la taille des listes augmentent.

Il est important, lorsque cela est possible, de connaître le nombre de clés afin de choisir au mieux la fonction de hachage et la méthode de résolution de collision à utiliser.

Cependant, ces méthodes dépendent du nombre de clés mais également de la taille m de la table. Il peut être nécessaire de modifier cette taille pour améliorer le comportement de la table de hachage.

Facteur de charge

Quand modifier la taille ?

Le taux d'occupation de la table doit rester relativement faible pour minimiser le risque de collision sinon le nombre de collision augmente et la performance se dégrade.

On définit le facteur de charge τ, comme le rapport entre le nombre d'alvéole utilisées dans la table et le nombre d'alvéole disponibles :

$\tau = \frac{n_a}{m}$ ou n_a est le nombre d'alvéole déjà utilisées et m la taille de la table.

Remarques :

Des facteurs de charge faibles n'induisent pas nécessairement de bonne performances, en particulier si la fonction de hachage génère des clusters.

L'augmentation de la taille de la table n'est pas trivial.