

TP - fonction de hachage minimale parfaite

Un algorithme inspiré de :

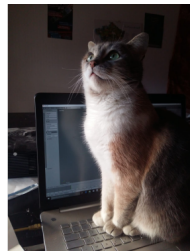
Fast and scalable minimal perfect hashing for massive key sets

[PDF] [arxiv.org](#)

[A Limasset](#), [G Rizk](#), [R Chikhi](#), [P Peterlongo](#) - arXiv preprint arXiv ..., 2017 - [arxiv.org](#)

... We provide a parallel C++ implementation called **BBhash**. It is capable of creating a minimal perfect hash function of 1010 elements in less than 7 minutes using 8 threads and 5 GB of ...

★ Enregistrer Citer Cité 109 fois Autres articles Les 15 versions



TP - fonction de hachage minimale parfaite

clefs $\{a,b,c,d,e,f,g\}$, ensemble de taille n
paramètres:

nombre de niveaux D (ici $D=2$)

facteur de taille gamma (ici gamma = 2)

1er niveau

créer le 1er tableau de taille gamma x n ,
y hasher les clefs

récolter les collisions $\{d,f,g\}$, ensemble de taille m

c	d, g × (collision)
a	f × (collision)
e	
b	

TP - fonction de hachage minimale parfaite

clefs $\{d, f, g\}$, ensemble de taille m

2e niveau

créer le 2e tableau de taille $\gamma \times m$

y hasher les clefs

**récolter les collisions $\{g\}$
à l'issue du dernier niveau**

c
a
e
b

f
d

$g \times$ (collision)

TP - fonction de hachage minimale parfaite

concaténer les tableaux
des différents niveaux

calculer les rangs des éléments

pour les dernières collisions,
donner le prochain rang disponible

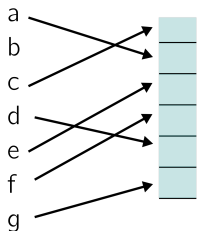
g -> **6**

rangs	
c	0
a	1
e	2
b	3
f	4
d	5

TP - fonction de hachage minimale parfaite

la MPHF est une fonction, c'est un objet qui retient:
(c,0) (a,1) (e,2) (b,3) (f,4) (d,5) (g,6)

Maintenant si j'alloue un tableau de taille n:



je peux parfaitement et minimalement
adresser mes clefs

TP - fonction de hachage minimale parfaite

En partie 1 du TP, on va l'implémenter !

Récupérer github.com/kamimrcht/enseignement/tree/main/TP2_hachage

Et voir la partie 1.