BJKST_enonce

April 17, 2019

1 1A.algo - BJKST - calculer le nombre d'éléments distincts

Comment calculer le nombre d'éléments distincts d'un ensemble de données quand celui-ci est trop grand pour tenir en mémoire. C'est ce que fait l'algorithme BJKST.

```
In [1]: from jyquickhelper import add_notebook_menu
        add_notebook_menu()
```

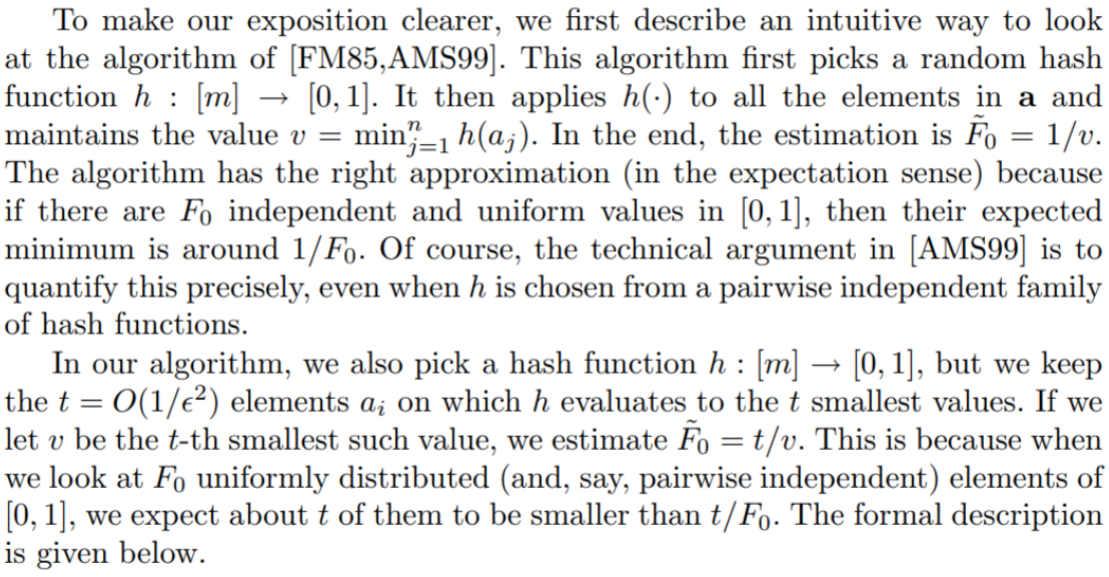
```
Out[1]: <IPython.core.display.HTML object>
```

1.1 Exercice 1 : première version

L'extrait qui suit est tiré de Counting distinct elements in a data stream. Il faut implémenter l'idée développé dans le second paragraphe.

```
In [2]: from pyquickhelper.helpgen import NbImage
        NbImage("images/bjkst.png", width=600)
```

```
Out[2]:
```



To make our exposition clearer, we first describe an intuitive way to look at the algorithm of [FM85,AMS99]. This algorithm first picks a random hash function $h : [m] \to [0, 1]$. It then applies $h(\cdot)$ to all the elements in $\mathbf{a}$ and maintains the value $v = \min_{j=1}^{n} h(a_j)$. In the end, the estimation is $\tilde{F}_0 = 1/v$. The algorithm has the right approximation (in the expectation sense) because if there are F_0 independent and uniform values in $[0, 1]$, then their expected minimum is around $1/F_0$. Of course, the technical argument in [AMS99] is to quantify this precisely, even when h is chosen from a pairwise independent family of hash functions.

In our algorithm, we also pick a hash function $h : [m] \to [0, 1]$, but we keep the $t = O(1/\epsilon^2)$ elements a_i on which h evaluates to the t smallest values. If we let v be the t-th smallest such value, we estimate $\tilde{F}_0 = t/v$. This is because when we look at F_0 uniformly distributed (and, say, pairwise independent) elements of $[0, 1]$, we expect about t of them to be smaller than t/F_0. The formal description is given below.

Saurez-vous convertir ce texte en un algorithme ?

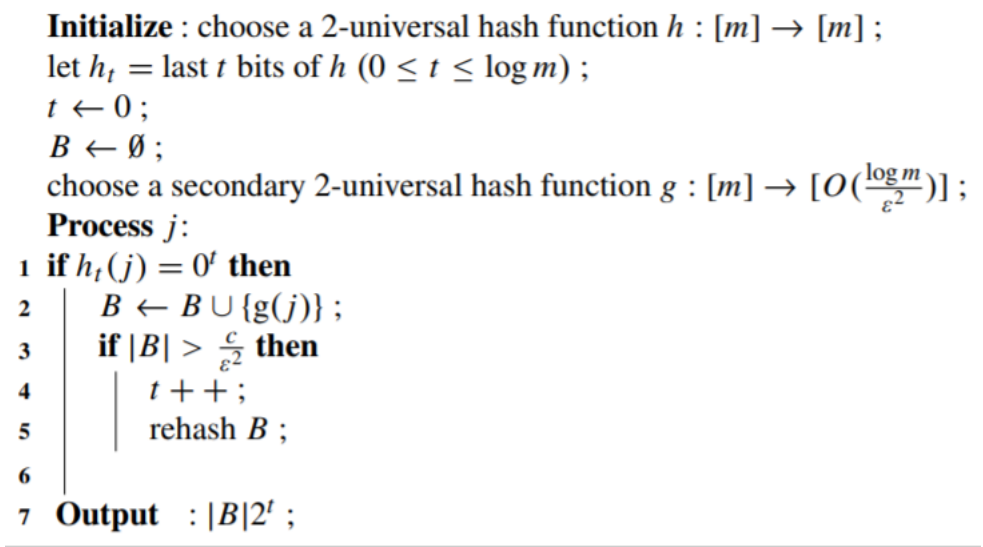
1.2 Exercice 2 : version plus rapide

L'extrait qui suit est tiré de CS85: Data Stream Algorithms, Lecture Notes (page 17). Il faut implémenter l'idée développé dans le second paragraphe.

In [3]:
```
from pyquickhelper.helpgen import NbImage
NbImage("images/bjkst2.png", width=600)
```

```
c:\python36_x64\lib\site-packages\sphinx\util\compat.py:40: RemovedInSphinx17Warning: sphinx.util.compat
  RemovedInSphinx17Warning)
```

Out[3]:



In [4]: