Algorithmique et Programmation

TD n° 3 : Hashing

École normale supérieure – Département d'informatique
algoL3@di.ens.fr

2016-2017

Some useful inequalities :
Bounds on $\binom{n}{k}$

$$\frac{n^k}{k^k} \leq \binom{n}{k} \leq \frac{n^k}{k!} \leq \frac{(en)^k}{k!}$$

$$k! \geq k^{k/2}$$

Union bound :

$$\mathbf{Pr}[A \cup B] \leq \mathbf{Pr}[A] + \mathbf{Pr}[B]$$

(with equality when A and B are disjoint.)

Exercise 1. Linear function

Let p be a prime and $h_{a,b}(i) = ai + b \bmod p$ a hash function. To show that if a and b are chosen uniformly at random from $0, 1, 2, \ldots, p-1$ and $p > n^2$, then with probability 1/2, there are no collision for n items $i_1, \ldots, i_n$.

Exercise 2. Hashing with chaining

Let $\mathcal{U}$ be a universe, $S \subset \mathcal{U}$ and m an integer.

1. Consider a hash table with chaining for the set S of cardinality $\#S = n$ built with a hash function h drawn uniformly at random among all the functions $\mathcal{U} \longrightarrow \{0, 1, \ldots, m-1\}$. Show that for $m = n$, the length of the longest linked list in the hash table is on the order of $O(\log n / \log\log n)$ with probability at least $1 - n^{-1}$ (with respect to the choice of hash functions).
2. (*) Suppose now that we use two hash functions h_1 and h_2 drawn independently and uniformly at random among all the functions $\mathcal{U} \longrightarrow \{0, 1, \ldots, m-1\}$ and an element x is inserted in the hash table at either $h_1(x)$ or $h_2(x)$ depending on which has the fewest elements at that time. Show that the length of the longest linked list in on the order of $O(\log\log n)$ with good probability.

Exercice 3. Cuckoo hash functions

Let U be a universe, $S \subset \mathcal{U}$ and m an integer. We use two hash functions h_1 and h_2 drawn independently and uniformly among all the functions $\mathcal{U} \longrightarrow \{0, 1, \ldots, m-1\}$. To insert an element x in the hash table, we calculate $h_1(x)$ and $h_2(x)$ and if one of the two positions is empty, we put x there. Otherwise, we remove the element y at $h_1(x)$, put x at $h_1(x)$ and put y in its other possible position. If this position is occupied by z, z is moved to its alternate position and so on. If the process fails (i.e. if we move the same element twice), we try to put x at position $h_2(x)$ in the same way.

If this process fails, then the entire hash table is rebuilt (by choosing two new new hash functions and re-inserting all the elements in the new table). We suppose that $n = \#S = m/4$.

1. Give the worst case complexity for the removal and search operations.
2. Consider the graph (called *cuckoo graph*) whose vertices are $V = \{0, \dots, M-1\}$ and edges are the pairs $\{h_1(x), h_2(x)\}$ for $x \in S$. Show that the hash table is reconstructed (if we try to insert all elements of S) if and only if there is a set of k vertices with at least $k+1$ edges between these vertices.
3. Show that if there is a set of k vertices with at least $k+1$ edges between them in a graph, then there exists a cycle in the graph. (A cycle is a sequence of vertices $v_1, \dots, v_\ell$ where all consecutive vertices are adjacent and v_1 is adjacent to v_ℓ.)
4. Show that the probability of having at least one edge between two fixed vertices v_1 and v_2 is at most $\frac{1}{2m}$. Where there is at least one edge, is the probability of having an edge between another pair is lower or greater ?

   Find the probability of having an edge between a vertex and itself.
5. Show that the probability the sequence $v_1, \dots, v_\ell$ forms a cycle is at most $\frac{1}{(2m)^\ell}$.
6. Show that the probability of the event in question 2 is at most by 1/2.
7. Deduce from this that the amortized cost of the insertion operation is $O(1)$ in expectation.