DICTIONARIES

1. Introduction

In this notes we introduce dictionary datatype and describe various solutions to dictionary problem and its variant. We mostly follow appropriate chapter of [Cormen et al., 2001], although there are two major differences. On the positive side our presentation is more mathematically rigorous. On the other hand some complexity estimations that we present are more rough.
After introducing dictionary datatype we discuss hash functions and analyze hash tables with chaining in the context of simple uniform hashing and universal families of hash functions. Next we introduce restricted dictionary problem and open addressing. We analyze open addressing under the assumption of uniform hashing. In the second part we introduce and analyze alternative family of solutions of dictionary problem, which are related to binary search trees and their balanced variants. In this part we discuss classical binary search trees and AVL trees. We supplement these approaches with careful mathematical analysis of their efficiency.

2. Dictionary data type

Definition 2.1. Let $\mathcal{X}$ be a set of *items* and let $\mathcal{U}$ be a set of *keys*. Consider an abstract data type D which dynamically stores a collection of pairs (k, x) where $k \in \mathcal{U}$ and $x \in \mathcal{X}$ in such a way that D does not store two pairs having the same key at the same time. Moreover, we assume that D supports the following operations.

INSERT$\big((k, x)\big)$
Adds (k, x) into D if there is no other pair stored in D with k as a first entry.

DELETE$\big(k\big)$
Removes a pair with k as a first entry from D if such pair is stored in D.

SEARCH$\big(k\big)$
Returns x if a pair (k, x) is stored in D. Otherwise returns *nil*.

An abstract data type with these properties and interface is called *an associative array* or *a dictionary*.

Definition 2.2. Let $\mathcal{X}$ and $\mathcal{U}$ be sets. *Dictionary problem for $\mathcal{X}$ and $\mathcal{U}$* is the task of designing a dictionary with $\mathcal{X}$ as the set of items and $\mathcal{U}$ as the set of keys.

3. Hash functions and hash tables with chaining

In this section we introduce the important notion of a hash function and we use it to solve dictionary problem.

Definition 3.1. Let $\mathcal{U}$ be a set. *A hash function* is a mapping $h : \mathcal{U} \to \{0, 1, ..., m-1\}$ where $m \in \mathbb{N}_+$.

Definition 3.2. Let $h : \mathcal{U} \to \{0, 1, ..., m-1\}$ be a hash function. *A collision* is a pair of keys $k_1, k_2 \in \mathcal{U}$ such that $k(k_1) = h(k_2)$.

Using hash functions one can solve dictionary problem. We introduce the notion which describes this solution.

Definition 3.3. Let $\mathcal{U}$ and $\mathcal{X}$ be sets. Let $h : \mathcal{U} \to \{0, 1, ..., m-1\}$ be a hash function for some $m \in \mathbb{N}_+$. We consider an m-element array D_h such that $D_h[l]$ is a linked list storing values from $\mathcal{U} \times \mathcal{X}$ for every $l \in \{0, 1, ..., m-1\}$. We describe dictionary operations.

$\mathrm{INSERT}_h\big((k,x)\big)$
Searches for a pair with the first entry k in the list $D_h[h(k)]$. If such pair is found, then replaces its second entry with x. If such pair is not found, then inserts pair (k,x) to the linked list $D_h[h(k)]$ as its new head.

$\mathrm{DELETE}_h\big(k\big)$
Deletes a pair with first entry k from the linked list $D_h[h(k)]$.

$\mathrm{SEARCH}_h\big(k\big)$
Searches for the pair with the first entry k in the list $D_h[h(k)]$. If such pair is found, then returns its second entry. Otherwise returns *nil*.

Then D_h together with these operations is a solution of dictionary problem for $\mathcal{U}$ and $\mathcal{X}$. We call it *the hash table with collisions resolved by chaining for h*.

4. Analysis of hash tables with chaining for simple uniform hashing

We start by introducing important stochastic property of hash functions.

Definition 4.1. Let $\mathcal{U}$ be a measurable space and let $h : \mathcal{U} \to \{0,1,...,m-1\}$ be a measurable hash function for some $m \in \mathbb{N}_+$. Suppose that μ is a probability distribution on $\mathcal{U}$. Fix a probability space $(\Omega, \mathcal{F}, P)$ and a sequence of independent random variables $K_1, ..., K_n : \Omega \to \mathcal{U}$ with distribution μ for some $n \in \mathbb{N}_+$. Consider the following assertions.

(1) Event
$$\mathcal{K} = \big\{\forall_{i,j\in\{1,...,n\},i\neq j}\, K_i \neq K_j\big\}$$
is of positive probability.

(2) Let $K : \Omega \to \mathcal{U}$ be a random variable with distribution μ and independent of $K_1, ..., K_n$. Then
$$P\big(h(K) = h(K_i)\,\big|\,\mathcal{K}\big) = \frac{1}{m}$$
for every $i \in \{1,...,n\}$.

If assertions above hold for every probability space $(\Omega, \mathcal{F}, P)$, every $n \in \mathbb{N}_+$ and every sequence $K_1, ..., K_n : \Omega \to \mathcal{U}$ of independent random variables with distribution μ, then h is *a simple uniform hashing with respect to μ*.

Now let us give two examples of hash functions satisfying simple uniform hashing with respect to canonical probability distributions on their spaces of keys.

Example 4.2. Let $\mathcal{U} = [0,m]$ for some $m \in \mathbb{N}_+$. Then $\mathcal{U}$ is a measurable space with respect to Borel algebra $\mathcal{B}\big([0,m]\big)$. We define a hash function $h : \mathcal{U} \to \{0,1,...,m-1\}$ by formula
$$h(x) = \lfloor x \rfloor$$
Then h is a simple uniform hashing with respect to the normalization of Lebesgue measure on $[0,m]$.

Example 4.3. Let $\mathcal{U} = \{0,1,...,m^2-1\}$ for some $m \in \mathbb{N}_+$. Then $\mathcal{U}$ is a measurable space with respect to the power algebra $\mathcal{P}\big(\{0,1,...,m^2-1\}\big)$. Consider $\mathcal{U}$ as a probability space with respect to the uniform distribution μ. We define a hash function $h : \mathcal{U} \to \{0,1,...,m-1\}$ by formula
$$h(x) = x \bmod m$$
We verify that h is a simple uniform hashing with respect to μ. Fix a probability space $(\Omega, \mathcal{F}, P)$ and $n \in \mathbb{N}_+$. Suppose first that $K_1, ..., K_n : \Omega \to \mathcal{U}$ are independent random variables with distribution μ. If
$$\mathcal{K} = \big\{\forall_{i,j\in\{1,...,n\},i\neq j}\, K_i \neq K_j\big\}$$

then

$$P(\mathcal{K}) = \frac{m^2 \cdot (m^2-1) \cdot \ldots \cdot (m^2-n+1)}{m^{2n}} > 0$$

Next suppose that $K : \Omega \to \mathcal{U}$ is a random variable with distribution μ which is independent of $K_1, \ldots, K_n$. Then for fixed $i \in \{1, \ldots, n\}$ we have

$$P\big(h(K) = h(K_i) \,\big|\, \mathcal{K}\big) = \frac{P\big(\{h(K) = h(K_i)\} \cap \mathcal{K}\big)}{P(\mathcal{K})} =$$

$$= m \cdot \frac{m^2 \cdot (m^2-1) \cdot \ldots \cdot (m^2-n+1)}{m^{2n+2}} \cdot \left(\frac{m^2 \cdot (m^2-1) \cdot \ldots \cdot (m^2-n+1)}{m^{2n}} \right)^{-1} = \frac{1}{m}$$

This completes the verification that h is a simple uniform hashing with respect to μ.

In order to analyze expected costs of dictionary operations for hash tables with chaining we introduce natural probabilistic model.

Setup 4.4 (Probabilistic model for hash tables with chaining)**.** We fix a measurable space of keys $\mathcal{U}$ and a set $\mathcal{X}$ of items. We consider a measurable hash function $h : \mathcal{U} \to \{0, 1, \ldots, m-1\}$ and a probability distribution μ on $\mathcal{U}$. We also fix a probability space $(\Omega, \mathcal{F}, P)$ and $n \in \mathbb{N}_+$. Let $K_1, \ldots, K_n : \Omega \to \mathcal{U}$ be independent random variables with distribution μ. Write

$$\mathcal{K} = \left\{ \forall_{i,j \in \{1,\ldots,n\}, i \neq j} \, K_i \neq K_j \right\}$$

and suppose that $K : \Omega \to \mathcal{U}$ is a random variable with distribution μ and independent from $K_1, \ldots, K_n$. We assume that pairs with keys $K_1(\omega), \ldots, K_n(\omega)$ for $\omega \in \mathcal{K}$ were consecutively inserted into initially empty D_h. Under this assumption we denote by $\mathbf{search}_h(K)$ the function $\mathcal{K} \to \mathbb{N}$ which for every $\omega \in \mathcal{K}$ returns the cost (in terms of the number of basic operations) of procedure

$$\mathrm{SEARCH}_h(K(\omega))$$

Similarly for procedure

$$\mathrm{DELETE}_h(K(\omega))$$

and (for fixed $x \in \mathcal{X}$) procedure

$$\mathrm{INSERT}_h\left((K(\omega), x)\right)$$

we define functions $\mathbf{delete}_h(K) : \mathcal{K} \to \mathbb{N}$ and $\mathbf{insert}_h\big((K, x)\big) : \mathcal{K} \to \mathbb{N}$.

In the remaining part of this section we work under probabilistic model described in Setup 4.4. We have the following fundamental result.

Theorem 4.5. *Let h be a simple uniform hashing with respect to μ. Then $\mathbf{search}_h(K) : \mathcal{K} \to \mathbb{N}$ is measurable and*

$$\mathbb{E}\, \mathbf{search}_h(K) = \int_{\mathcal{K}} \mathbf{search}_h(K) \, dP_{\mathcal{K}} \leq 1 + \frac{n}{m}$$

Proof. First we introduce certain notation. We consider events

$$W_i = \big\{h\left(K_i\right) = h\left(K\right)\big\}, \; Z_i = \big\{K = K_i\big\} \cap \mathcal{K}, \; Z = \bigcup_{i=1}^{n} Z_i$$

for $i \in \{1, \ldots, n\}$. Fix $\omega \in \mathcal{K}$. For the procedure

$$\mathrm{SEARCH}_h(K(\omega))$$

we first calculate $h(K(\omega))$. This is a single basic operation. Next if $\omega \in \mathcal{K} \setminus Z$, then we iterate through the list $D_h[h(K(\omega))]$ with length equal to the number of keys in $\{K_1(\omega), \ldots, K_n(\omega)\}$ mapped by h to $h(K(\omega))$. This is the case of the unsuccessful search. Otherwise, if $\omega \in Z_i$ then we iterate through the initial segment of the list $D_h[h(K(\omega))]$ which consists of elements from

the set $\{K_i(\omega), K_{i+1}(\omega), ..., K_n(\omega)\}$ mapped by h to $h(K(\omega))$. This is the case when the search is successful. Thus

$$\mathbf{search}_h(K) = \underbrace{1}_{\text{computation of the hash}} + \underbrace{\chi_{\mathcal{K}\setminus Z} \cdot \sum_{i=1}^{n} \chi_{W_i}}_{\text{unsuccessful search}} + \underbrace{\sum_{i=1}^{n} \chi_{Z_i} \cdot \sum_{j=i}^{n} \chi_{W_j}}_{\text{succesfull search}}$$

Hence $\mathbf{search}_h(K) : \mathcal{K} \to \mathbb{N}$ is measurable. Moreover, note that

$$\mathbf{search}_h(K) = 1 + \chi_{\mathcal{K}\setminus Z} \cdot \sum_{i=1}^{n} \chi_{W_i} + \sum_{i=1}^{n} \chi_{Z_i} \cdot \sum_{j=i}^{n} \chi_{W_j} \leq 1 + \chi_{\mathcal{K}\setminus Z} \cdot \sum_{i=1}^{n} \chi_{W_i} + \sum_{i=1}^{n} \chi_{Z_i} \cdot \sum_{j=1}^{n} \chi_{W_j} = 1 + \sum_{i=1}^{n} \chi_{W_i}$$

and hence

$$\mathbb{E}\,\mathbf{search}_h(K) \leq 1 + \sum_{i=1}^{n} \mathbb{E}\,\chi_{W_i} = 1 + \sum_{i=1}^{n} P_{\mathcal{K}}(W_i) = 1 + \sum_{i=1}^{n} \frac{P(W_i \cap \mathcal{K})}{P(\mathcal{K})} = 1 + \frac{n}{m}$$

The last inequality is a consequence of the fact that h is a simple uniform hashing with respect to μ. □

Using essentially the same method (we omit the proof) one derives the following results.

Theorem 4.6. *Let h be a simple uniform hashing with respect to μ. Fix x in $\mathcal{X}$. Then both functions $\mathbf{delete}_h(K), \mathbf{insert}_h((K,x)) : \mathcal{K} \to \mathbb{N}$ are measurable and*

$$\mathbb{E}\,\mathbf{delete}_h(K), \mathbb{E}\,\mathbf{insert}_h((K,x)) \leq 1 + \frac{n}{m}$$

These results have the following consequence.

Corollary 4.7. *Let h be a simple uniform hashing with respect to μ. Suppose that there exists a constant $c \in \mathbb{R}_+$ such that $n \leq c \cdot m$. Then the expected costs of all dictionary operations for D_h are $\mathcal{O}(1)$.*

Proof. The assertion follows from Theorems 4.5 and 4.6 and the inequality $1 + \frac{n}{m} \leq 1 + c$. □

5. Universal hash function families

In this section we introduce and study important notion of universal hash function family. We start by proving the result, which gives lower bound on probability of key collisions.

Proposition 5.1. *Let $\mathcal{U}$ be a finite set and let $m \in \mathbb{N}_+$. Suppose that $\mathcal{H}$ is a finite family of functions $\mathcal{U} \to \{0, 1, ..., m-1\}$ and let P be a uniform probability distribution on $\mathcal{H}$. Assume that for some $\epsilon \geq 0$ and for every pair of distinct elements $k, l \in \mathcal{U}$ we have*

$$P\big(h(k_1) = h(k_2)\big) \leq \epsilon$$

Then

$$\epsilon \geq \frac{1}{m} - \frac{1}{|\mathcal{U}|}$$

Proof. For each pair of distinct $k, l \in \mathcal{U}$ consider the indicator random variable $X_{k,l}$ of the subset $\{h \in \mathcal{H} \,|\, h(k) = h(l)\} \subseteq \mathcal{H}$. Note that

$$\sum_{k,l \in \mathcal{U},\, k \neq l} X_{k,l}(h) = \text{the total number of collisions for a hash function } h : \mathcal{U} \to \{0, 1, ..., m-1\}$$

for every $h \in \mathcal{H}$. Now fix an arbitrary hash function $h : \mathcal{U} \to \{0, 1, ..., m-1\}$. For every $i \in \{0, 1, ..., m-1\}$ let $h_i = |h^{-1}(i)|$. Then the number of collisions for this particular function is given

by

$$\sum_{i=0}^{m-1}\binom{h_i}{2} = \frac{1}{2}\cdot\left(\sum_{i=0}^{m-1} h_i^2 - \sum_{i=0}^{m-1} h_i\right) \geq \frac{1}{2}\cdot\frac{\left(\sum_{i=0}^{m-1} h_i\right)^2}{m} - \frac{1}{2}\cdot\sum_{i=0}^{m-1} h_i = \frac{1}{2}\cdot\left(\frac{|\mathcal{U}|^2}{m} - |\mathcal{U}|\right)$$

Thus we derive that

$$\sum_{k,l\in\mathcal{U},\, k\neq l} X_{k,l} \geq \frac{1}{2}\cdot\left(\frac{|\mathcal{U}|^2}{m} - |\mathcal{U}|\right)$$

Taking expectations with respect to P we obtain

$$\binom{|\mathcal{U}|}{2}\cdot\epsilon \geq \sum_{k,l\in\mathcal{U},\, k\neq l} \mathbb{E}\, X_{k,l} = \mathbb{E}\sum_{k,l\in\mathcal{U},\, k\neq l} X_{k,l} \geq \frac{1}{2}\cdot\left(\frac{|\mathcal{U}|^2}{m} - |\mathcal{U}|\right)$$

and hence

$$\epsilon \geq \frac{\frac{1}{2}\cdot\left(\frac{|\mathcal{U}|^2}{m} - |\mathcal{U}|\right)}{\binom{|\mathcal{U}|}{2}} \geq \frac{\frac{1}{2}\cdot\left(\frac{|\mathcal{U}|^2}{m} - |\mathcal{U}|\right)}{\frac{1}{2}\cdot|\mathcal{U}|^2} = \frac{\frac{|\mathcal{U}|^2}{m} - |\mathcal{U}|}{|\mathcal{U}|^2} = \frac{1}{m} - \frac{1}{|\mathcal{U}|}$$

□

Definition 5.2. Let $\mathcal{U}$ be a set and let $m \in \mathbb{N}_+$. Suppose that $\mathcal{H}$ is a finite family of functions $\mathcal{U} \to \{0, 1, ..., m-1\}$ and let P be a uniform probability distribution on $\mathcal{H}$. Suppose that

$$P\big(h(k_1) = h(k_2)\big) \leq \frac{1}{m}$$

for every pair of distinct keys $k_1, k_2 \in \mathcal{U}$. Then $\mathcal{H}$ is *a universal family of hash function*.

Note that Proposition 5.1 asserts that universal families of hash functions are optimal.
The remaining part of this section is devoted to giving examples of universal families of hash functions.

Example 5.3. Let p be a prime number. Consider $\mathcal{U} = \mathbb{Z}_p^n$ for some $n \in \mathbb{N}_+$. We have bilinear map

$$\mathcal{U}\times\mathcal{U} \ni (k,l) \mapsto \langle k,l\rangle \in \mathbb{Z}_p$$

where

$$\langle k,l\rangle = \sum_{i=1}^{n} k_i\cdot l_i$$

and $k = (k_1, ..., k_n)$, $l = (l_1, ...l_n)$ are coordinate expansions. Now we define the family of functions

$$\mathcal{H} = \left\{h : \mathcal{U} \to \mathbb{Z}_p \,\middle|\, \exists_{a\in\mathbb{Z}_p}\ \forall_{k\in\mathcal{U}}\ h(k) = \langle a,k\rangle\right\}$$

We show now that $\mathcal{H}$ is a family of universal hash functions. For this pick two distinct keys $k, l \in \mathcal{U}$. We have

$$P\big(h(k) = h(l)\big) = \frac{\left|\left\{a \in \mathbb{Z}_p^n \,\middle|\, \langle a,k\rangle = \langle a,k\rangle\right\}\right|}{|\mathbb{Z}_p^n|} = \frac{\left|\left\{a \in \mathbb{Z}_p^n \,\middle|\, \langle a,k-l\rangle = 0\right\}\right|}{|\mathbb{Z}_p^n|} =$$

$$= \frac{\left|\ker\left(\mathbb{Z}_p^n \ni a \mapsto \langle a,k-l\rangle \in \mathbb{Z}_p\right)\right|}{|\mathbb{Z}_p^n|} = \frac{p^{n-1}}{p^n} = \frac{1}{p}$$

Example 5.4. Consider $\mathcal{U} = \mathbb{Z}_p$ for some prime number p and suppose that $m < p$ for some $m \in \mathbb{N}_+$. We define

$$\mathcal{H} = \left\{h : \mathcal{U} \to \{0, 1, ..., m-1\} \,\middle|\, \exists_{a\in\mathbb{Z}_p^*}\ \exists_{b\in\mathbb{Z}_p}\ \forall_{k\in\mathcal{U}}\ h(k) = (a\cdot k + b \bmod p) \bmod m\right\}$$

We show now that $\mathcal{H}$ is a family of universal hash functions. For this pick two distinct keys $k, l \in \mathcal{U}$ and note that the function

$$\mathbb{Z}_p^*\times\mathbb{Z}_p \ni (a,b) \mapsto a\cdot(k-l) + b \bmod p \in \mathbb{Z}_p$$

has the property that its fiber over each element in $\mathbb{Z}_p$ has exactly $(p-1)$-elements. We have

$$P\big(h(k)=h(l)\big)=\frac{\big|\{(a,b)\in\mathbb{Z}_p^*\times\mathbb{Z}_p\;\big|\;(a\cdot k+b \bmod p) \bmod m=(a\cdot l+b \bmod p) \bmod m\}\big|}{|\mathbb{Z}_p^*\times\mathbb{Z}_p|}=$$

$$=\frac{\big|\{(a,b)\in\mathbb{Z}_p^*\times\mathbb{Z}_p\;\big|\;(a\cdot(k-l)+b \bmod p) \bmod m=0\}\big|}{|\mathbb{Z}_p^*\times\mathbb{Z}_p|}=\frac{(p-1)\cdot\lfloor\frac{p}{m}\rfloor}{(p-1)\cdot p}\leq\frac{(p-1)\cdot\frac{p}{m}}{(p-1)\cdot p}=\frac{1}{m}$$

6. Analysis of hash tables with chaining for universal families of hash functions

In this section we describe alternative analysis of hash tables with chaining, which is based on the notion of universal family of hash functions.

Setup 6.1 (Probabilistic model for hash tables with chaining)**.** Let $\mathcal{U},\mathcal{X}$ be sets and let $m\in\mathbb{N}_+$. Suppose that $\mathcal{H}$ is a finite family of functions $\mathcal{U}\to\{0,1,...,m-1\}$ and let P be a uniform probability distribution on $\mathcal{H}$. We fix $n\in\mathbb{N}_+$ and distinct keys $k_1,...,k_n\in\mathcal{U}$. For each $h\in\mathcal{H}$ we assume that pairs with keys $k_1,...,k_n$ were consecutively inserted into initially empty D_h. We also fix $k\in\mathcal{U}$. Under this assumption we denote by $\mathbf{search}_{\mathcal{H}}(k)$ the function $\mathcal{H}\to\mathbb{N}$ which for every $h\in\mathcal{H}$ returns the cost (in terms of the number of basic operations) of procedure

$$\mathrm{SEARCH}_h(k)$$

Similarly for procedure

$$\mathrm{DELETE}_h(k)$$

and (for fixed $x\in\mathcal{X}$) procedure

$$\mathrm{INSERT}_h\left((k,x)\right)$$

we define functions $\mathbf{delete}_{\mathcal{H}}(K):\mathcal{H}\to\mathbb{N}$ and $\mathbf{insert}_{\mathcal{H}}\big((k,x)\big):\mathcal{H}\to\mathbb{N}$.

In the remaining part of this section we work under probabilistic model described in Setup 6.1. We have the following fundamental result.

Theorem 6.2. *Let $\mathcal{H}$ be a universal family of hash functions and let k be a key in $\mathcal{U}$. Then* $\mathbf{search}_{\mathcal{H}}(k):\mathcal{H}\to\mathbb{N}$ *is measurable and*

$$\mathbb{E}\,\mathbf{search}_{\mathcal{H}}(k)=\int_{\mathcal{H}}\mathbf{search}_{\mathcal{H}}(k)\,dP\leq 1+\frac{n}{m}$$

Proof. First we introduce certain notation. We consider events $W_i=\big\{h(k_i)=h(k)\big\}$ for $i\in\{1,...,n\}$. For the procedure

$$\mathrm{SEARCH}_h(k)$$

we first calculate $h(k)$. This is a single basic operation. Next if k is not one of the keys $k_1,...,k_n$, then we iterate through the list $D_h[h(k)]$ with length equal to the number of keys in $\big\{k_1,...,k_n\big\}$ mapped by h to $h(k)$. This is the case of the unsuccessful search. Hence

$$\mathbf{search}_{\mathcal{H}}(k)=\underbrace{1}_{\text{computation of the hash}}+\underbrace{\sum_{i=1}^{n}\chi_{W_i}}_{\text{unsuccessful search}}$$

If on the other hand k is equal to say k_i for some $i\in\{1,...,n\}$, then we iterate through the initial segment of the list $D_h[h(k)]$ which consists of elements from the set $\big\{k_i,k_{i+1},...,k_n\big\}$ mapped by h to $h(k)$. This is the case when the search is successful. Hence

$$\mathbf{search}_{\mathcal{H}}(k)=\underbrace{1}_{\text{computation of the hash}}+\underbrace{\sum_{j=i}^{n}\chi_{W_j}}_{\text{succesfull search}}$$

In either case $\mathbf{search}_{\mathcal{H}}(k) : \mathcal{H} \to \mathbb{N}$ is measurable. Moreover, note that

$$\mathbf{search}_{\mathcal{H}}(k) \leq 1 + \sum_{i=1}^{n} \chi_{W_i}$$

for every $k \in \mathcal{U}$ and thus

$$\mathbb{E}\,\mathbf{search}_{\mathcal{H}}(k) \leq 1 + \sum_{i=1}^{n} \mathbb{E}\,\chi_{W_i} = 1 + \sum_{i=1}^{n} P(W_i) = 1 + \frac{n}{m}$$

The last inequality is a consequence of the fact that $\mathcal{H}$ is a universal family of hash functions. □

Using essentially the same method (we omit the proof) one derives the following results.

Theorem 6.3. *Let $\mathcal{H}$ be a universal family of hash functions and let k be a key in $\mathcal{U}$. Fix x in $\mathcal{X}$. Then both functions $\mathbf{delete}_{\mathcal{H}}(k), \mathbf{insert}_{\mathcal{H}}((k,x)) : \mathcal{H} \to \mathbb{N}$ are measurable and*

$$\mathbb{E}\,\mathbf{delete}_{\mathcal{H}}(k), \mathbb{E}\,\mathbf{insert}_{\mathcal{H}}((k,x)) \leq 1 + \frac{n}{m}$$

These results have the following consequence.

Corollary 6.4. *Let $\mathcal{H}$ be a universal family of hash functions. Suppose that there exists a constant $c \in \mathbb{R}_+$ such that $n \leq c \cdot m$. Then the expected costs of all dictionary operations for D_h are $\mathcal{O}(1)$ provided that h is taken uniformly random from $\mathcal{H}$.*

Proof. The assertion follows from Theorems 6.2 and 6.3 and the inequality $1 + \frac{n}{m} \leq 1 + c$. □

7. Open addressing and uniform hashing

In this section we consider only search and insert dictionary operations. Let us be more precise.

Definition 7.1. Let $\mathcal{X}$ be a set of *items* and let $\mathcal{U}$ be a set of *keys*. Consider an abstract data type D which dynamically stores a collection of pairs (k,x) where $k \in \mathcal{U}$ and $x \in \mathcal{X}$ in such a way that D does not store two pairs having the same key at the same time. Moreover, we assume that D supports the following operations.

INSERT$\big((k,x)\big)$
Adds (k,x) into D if there is no other pair stored in D with k as a first entry.

SEARCH$\big(k\big)$
Returns x if a pair (k,x) is stored in D. Otherwise returns *nil*.

An abstract data type with these properties and interface is called *a restricted associative array* or *a restricted dictionary*.

Definition 7.2. Let $\mathcal{X}$ and $\mathcal{U}$ be sets. *Restricted dictionary problem for $\mathcal{X}$ and $\mathcal{U}$* is the task of designing a restricted dictionary with $\mathcal{X}$ as the set of items and $\mathcal{U}$ as the set of keys.

Now we describe certain solution to a restricted dictionary problem.

Definition 7.3. Let $\mathcal{U}$ and $\mathcal{X}$ be sets. Let $h : \mathcal{U} \times \{0,1,...,m-1\} \to \{0,1,...,m-1\}$ be a function for some $m \in \mathbb{N}_+$. We consider an m-element array D_h such that $D_h[l]$ has the capacity to store value from $\mathcal{U} \times \mathcal{X}$ for each $l \in \{0,1,...,m-1\}$. We describe restricted dictionary operations.

INSERT$_h\big((k,x)\big)$
Iterate through the list

$$D[h(k,0)], D[h(k,1)], ..., D[h(k,m-1)]$$

and find either the first empty slot or a slot which stores a pair with k as the first entry. In case when some slot is found, then sets its value to (k, x). If there are no empty slots and no slots with k as the first entry, then raise appropriate exception.

SEARCH$_h(k)$
Iterate through the list

$$D[h(k,0)], D[h(k,1)], ..., D[h(k,m-1)]$$

and find either the first empty slot or a slot which stores a pair with k as the first entry. In case when empty slot is found return *nil*. If a slot with pair having k as the first entry is found, then return the second entry of the pair. If there are no empty slots and no slots with k as the first entry, then raise appropriate exception.

Then D_h together with these operations is a solution of restricted dictionary problem for $\mathcal{U}$ and $\mathcal{X}$. We call it *the hash table with open addressing for* h.

8. Analysis of hash tables with open addressing for uniform hashing

Let us start with generalization of simple uniform hashing.

Definition 8.1. Let $\mathcal{U}$ be a measurable space and let $h : \mathcal{U} \times \{0, 1, ..., m-1\} \to \{0, 1, ..., m-1\}$ be a measurable function for some $m \in \mathbb{N}_+$. Suppose that μ is a probability distribution on $\mathcal{U}$. Fix a probability space $(\Omega, \mathcal{F}, P)$ and a random variable $K : \Omega \to \mathcal{U}$ with distribution μ. Assume that

$$P\big(\forall_{l \in \{0,1,...,m-1\}}\, h(K, l) = \sigma(l)\big) = \frac{1}{m!}$$

for every permutation $\sigma : \{0, 1, ..., m-1\} \to \{0, 1, ..., m-1\}$. If this equality holds for every probability space $(\Omega, \mathcal{F}, P)$ and a random variable $K : \Omega \to \mathcal{U}$ with distribution μ, then h is *a uniform hashing with respect to* μ.

Now we discuss canonical example of uniform hashing.

Example 8.2. Fix $m \in \mathbb{N}_+$ and let $\mathcal{U}$ be the set of all permutations $\{0, 1, ..., m-1\} \to \{0, 1, ..., m-1\}$. Suppose that μ is a uniform distribution on $\mathcal{U}$. We define a function $h : \mathcal{U} \times \{0, 1, ..., m-1\} \to \{0, 1, ..., m-1\}$ by formula

$$h(k, i) = k(i)$$

for each k in $\mathcal{U}$ and each $i \in \{0, 1, ..., m-1\}$. Then h is a uniform hashing with respect to μ.

In order to analyze open addressing we introduce probabilistic model.

Setup 8.3 (Probabilistic model for hash tables with open addressing)**.** We fix a measurable space of keys $\mathcal{U}$ and a set $\mathcal{X}$ of items. We consider a measurable function $h : \mathcal{U} \times \{0, 1, ..., m-1\} \to \{0, 1, ..., m-1\}$ and a probability distribution μ on $\mathcal{U}$. We also fix a probability space $(\Omega, \mathcal{F}, P)$. Let $k_1, ..., k_n$ be distinct elements of $\mathcal{U}$ for some $n \in \mathbb{N}_+$ such that $n < m$. We make technical assumption that all subsets of $\{k_1, ..., k_n\}$ are measurable with respect to the σ-algebra of $\mathcal{U}$. We assume that pairs with keys $k_1, ..., k_n$ were consecutively inserted into initially empty D_h. Suppose that $K : \Omega \to \mathcal{U}$ is a random variable with distribution μ. Under this assumptions we denote by $\mathbf{search}_h(K)$ the cost (in terms of the number of basic operations) of procedure

$$\text{SEARCH}_h(K(\omega))$$

Similarly for fixed $x \in \mathcal{X}$ and procedure

$$\text{INSERT}_h\left((K(\omega), x)\right)$$

we define the function $\mathbf{insert}_h\big((K, x)\big) : \Omega \to \mathbb{N}$.

In the remaining part of this section we work under probabilistic model described in Setup 8.3. We have the following fundamental result.

Theorem 8.4. *Let h be a uniform hashing with respect to μ. Then* $\mathbf{search}_h(K) : \Omega \to \mathbb{N}$ *is measurable and*

$$\mathbb{E}\,\mathbf{search}_h(K) = \int_\Omega \mathbf{search}_h(K)\,dP \leq \frac{2}{1-\frac{n}{m}}$$

Proof. According to Setup 8.3 pairs with keys $k_1, ...k_n$ were consecutively inserted to initially empty D_h. Suppose that pair with key k_j occupy slot s_j for $j \in \{1, ..., n\}$. We define

$$W_i = \big\{\forall_{l\leq i-1}\, h(K,l) \in \{s_1, ..., s_n\} \text{ and } k(K,i) \notin \{s_1, ..., s_n\}\big\},\ Z_j = \big\{K = k_j\big\},\ Z = \bigcup_{j=1}^{n} Z_j$$

for $i \in \{0, ..., m-1\}$ and $j \in \{1, ..., n\}$. Note that all these sets are elements of $\mathcal{F}$. Fix $\omega \in \Omega$. We describe now the procedure

$$\mathrm{SEARCH}_h(K)$$

in different scenarios. If $\omega \in W_i \setminus Z$ for some $i \in \{0, 1, ..., m-1\}$, then the search is unsuccessful and we first iterate through slots indexed by

$$h\big(K(\omega), 0\big), ..., h\big(K(\omega), i-1\big)$$

which are occupied and then we found empty slot indexed by $h\big(K(\omega), i\big)$. Otherwise, if $\omega \in Z_j$ for some $j \in \{1, ..., n\}$, then there exists a unique $m_j \in \{0, 1, ..., m-1\}$ such that $h\big(K(\omega), m_j\big) = s_j$. In this case we iterate through slots indexed by

$$h\big(K(\omega), 0\big), ..., h\big(K(\omega), m_j - 1\big)$$

which are occupied and then then we found slot indexed by $h\big(K(\omega), m_j\big) = s_j$ which is occupied with a pair having $K(\omega) = k_j$ as the first entry. This is the case when the search is successful. Thus

$$\mathbf{search}_h\,(K) = 2 \cdot \Bigg(\underbrace{\chi_{\Omega\setminus Z} \cdot \sum_{i=0}^{m-1} \chi_{W_i}}_{\text{unsuccessful search}} + \underbrace{\sum_{j=1}^{n} m_j \cdot \chi_{Z_j}}_{\text{succesfull search}} \Bigg)$$

Hence $\mathbf{search}_h\,(K) : K \to \mathbb{N}$ is measurable. Moreover, from the description above it follows that

$$\mathbf{search}_h\,(K) \leq 2 \cdot \sum_{i=0}^{m-1} \chi_{W_i}$$

Since h is a uniform hashing with respect to μ, we derive that

$$\mathbb{E}\,\chi_{W_i} = \frac{n\cdot(n-1)\cdot ...\cdot(n-i+1)\cdot(m-n)\cdot(m-i-1)!}{m!} \leq \frac{n\cdot(n-1)\cdot ...\cdot(n-i+1)}{m\cdot(m-1)\cdot ...\cdot(m-i+1)} \leq \left(\frac{n}{m}\right)^i$$

for every $i \in \{0, 1, ..., m-1\}$. Hence

$$\mathbb{E}\,\mathbf{search}_h\,(K) \leq 2 \cdot \sum_{i=0}^{m-1} \mathbb{E}\,\chi_{W_i} \leq 2 \cdot \sum_{i=0}^{m-1} \left(\frac{n}{m}\right)^i \leq 2 \cdot \sum_{i=0}^{+\infty} \left(\frac{n}{m}\right)^i = \frac{2}{1-\frac{n}{m}}$$

□

Using essentially the same method (we omit the proof) one derives the following results.

Theorem 8.5. *Let h be a uniform hashing with respect to μ and fix x in $\mathcal{X}$. Then* $\mathbf{insert}_h\big((K,x)\big) : \Omega \to \mathbb{N}$ *is measurable and*

$$\mathbb{E}\,\mathbf{insert}_h\big((K,x)\big) = \int_\Omega \mathbf{insert}_h\big((K,x)\big)\,dP \leq \frac{2}{1-\frac{n}{m}}$$

These results have the following consequence.

Corollary 8.6. *Let h be a uniform hashing with respect to μ and fix x in $\mathcal{X}$. Suppose that there exists a constant $c \in (0,1)$ such that $n \leq c \cdot m$. Then the expected costs of all restricted dictionary operations for D_h are $\mathcal{O}(1)$.*

Proof. The assertion follows from Theorems 8.4 and 8.5 and the inequality

$$\frac{2}{1-\frac{n}{m}} \leq \frac{2}{1-c}$$

□

9. Binary Search Trees

Notation 9.1. Let T be a binary tree. We denote the root of T by $root(T)$. We also denote by $left(T)$ and $right(T)$ the left and, respectively, right subtrees of T.

Definition 9.2. Let $\mathcal{U}$ be a set and let $\mathcal{X}$ be a set. Consider a pair $(T, \mathrm{k}, \mathrm{v})$ such that T is a binary tree and $\mathrm{k} : V(T) \to \mathcal{U}, \mathrm{v} : V(T) \to \mathcal{X}$ are functions defined on the set of vertices $V(T)$ of T. Then $(T, \mathrm{k}, \mathrm{v})$ is called *a binary tree with keys in $\mathcal{U}$ and values in $\mathcal{X}$*.

Definition 9.3. Let $\mathcal{U}$ be a linearly ordered set and let $\mathcal{X}$ be a set. A binary tree $(T, \mathrm{k}, \mathrm{v})$ such that

$$\mathrm{k}(\mathrm{left}(v)) < \mathrm{k}(v) < \mathrm{k}(\mathrm{right}(v))$$

for every vertex v of T is called *a binary search tree with keys in $\mathcal{U}$ and values in $\mathcal{X}$*. We describe restricted dictionary operations on such a tree.

BST-INSERT$\big((k,x)\big)$
If $\mathrm{k}\big(root(T)\big) = k$, then set $\mathrm{v}\big(root(T)\big) = x$. If $left(T) = right(T) = \varnothing$ and

BST-SEARCH$\big(k\big)$
Iterate through the list

$$D[h(k,0)], D[h(k,1)], ..., D[h(k,m-1)]$$

and find either the first empty slot or a slot which stores a pair with k as the first entry. In case when empty slot is found return *nil*. If a slot with pair having k as the first entry is found, then return the second entry of the pair. If there are no empty slots and no slots with k as the first entry, then raise appropriate exception.

Then D_h together with these operations is a solution of restricted dictionary problem for $\mathcal{U}$ and $\mathcal{X}$. We call it *the hash table with open addressing for h*.

References

[Cormen et al., 2001] Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. (2001). *Introduction to algorithms*. MIT Press, Cambridge, MA; McGraw-Hill Book Co., Boston, MA, second edition.