

Algorithms and Datastructures

Hash Map, Universal Hashing

Prof. Dr. Rolf Backofen

Bioinformatics Group / Department of Computer Science

Algorithms and Datastructures, November 2017

Structure

Associative Arrays

- Introduction

- Hash Map

Universal Hashing

- Introduction

- Probability Calculation

- Proof

- Examples

Structure

Associative Arrays

- Introduction

- Hash Map

Universal Hashing

- Introduction

- Probability Calculation

- Proof

- Examples

Associative Arrays

How do we build a Map?

Reminder:

- ▶ An associative array is like a normal array, only that the indices are not $0, 1, 2, \dots$, but different, e.g. telephone numbers

Associative Arrays

How do we build a Map?

Reminder:

- ▶ An associative array is like a normal array, only that the indices are not $0, 1, 2, \dots$, but different, e.g. telephone numbers

Problem:

- ▶ Quickly find a element with a specific key

Associative Arrays

How do we build a Map?

Reminder:

- ▶ An associative array is like a normal array, only that the indices are not $0, 1, 2, \dots$, but different, e.g. telephone numbers

Problem:

- ▶ Quickly find a element with a specific key
- ▶ Naive solution: Store pairs of key and value in a normal field

Associative Arrays

How do we build a Map?

Reminder:

- ▶ An associative array is like a normal array, only that the indices are not $0, 1, 2, \dots$, but different, e.g. telephone numbers

Problem:

- ▶ Quickly find a element with a specific key
- ▶ Naive solution: Store pairs of key and value in a normal field
- ▶ For n keys searching requires $\Theta(n)$ time

Associative Arrays

How do we build a Map?

Reminder:

- ▶ An associative array is like a normal array, only that the indices are not $0, 1, 2, \dots$, but different, e.g. telephone numbers

Problem:

- ▶ Quickly find a element with a specific key
- ▶ Naive solution: Store pairs of key and value in a normal field
- ▶ For n keys searching requires $\Theta(n)$ time
- ▶ With a **hash map** this just requires $\Theta(1)$ in the best case, ... regardless how many elements are in the map!

Structure

Associative Arrays

Introduction

Hash Map

Universal Hashing

Introduction

Probability Calculation

Proof

Examples

Associative Arrays

The Hash Map

Idea:

- ▶ Mapping the keys onto indices with a **hash function**
- ▶ Store the values at the calculated indices in a normal array

Example:

- ▶ Key set: $x = \{3904433, 312692, 5148949\}$

Associative Arrays

The Hash Map

Idea:

- ▶ Mapping the keys onto indices with a **hash function**
- ▶ Store the values at the calculated indices in a normal array

Example:

- ▶ Key set: $x = \{3904433, 312692, 5148949\}$
- ▶ Hash function: $h(x) = x \bmod 5$, in the range $[0, \dots, 4]$

Associative Arrays

The Hash Map

Idea:

- ▶ Mapping the keys onto indices with a **hash function**
- ▶ Store the values at the calculated indices in a normal array

Example:

- ▶ Key set: $x = \{3904433, 312692, 5148949\}$
- ▶ Hash function: $h(x) = x \bmod 5$, in the range $[0, \dots, 4]$
- ▶ We need an array **T** with **5** elements.
A “hashtable” with 5 “buckets”

Associative Arrays

The Hash Map

Idea:

- ▶ Mapping the keys onto indices with a **hash function**
- ▶ Store the values at the calculated indices in a normal array

Example:

- ▶ Key set: $x = \{3904433, 312692, 5148949\}$
- ▶ Hash function: $h(x) = x \bmod 5$, in the range $[0, \dots, 4]$
- ▶ We need an array **T** with **5** elements.
A “hashtable” with 5 “buckets”
- ▶ The element with the key **x** is stored in $T[h(x)]$

Associative Arrays

The Hash Map

Storage:

Figure: Hashtable T

0	
1	
2	
3	
4	

Associative Arrays

The Hash Map

Storage:

- ▶ `insert(3904433, "A")`: $h(3904433) = 3 \Rightarrow T[3] = (3904433, "A")$

Figure: Hashtable T



Associative Arrays

The Hash Map

Storage:

- ▶ `insert(3904433, "A")`: $h(3904433) = 3 \Rightarrow T[3] = (3904433, "A")$
- ▶ `insert(312692, "B")`: $h(312692) = 2 \Rightarrow T[2] = (312692, "B")$

Figure: Hashtable T



Associative Arrays

The Hash Map

Storage:

- ▶ `insert(3904433, "A")`: $h(3904433) = 3 \Rightarrow T[3] = (3904433, "A")$
- ▶ `insert(312692, "B")`: $h(312692) = 2 \Rightarrow T[2] = (312692, "B")$
- ▶ `insert(5148949, "C")`: $h(5148949) = 4 \Rightarrow T[4] = (5148949, "C")$

Figure: Hashtable T



Associative Arrays

The Hash Map

Searching:

- ▶ `search(3904433): $h(3904433) = 3 \Rightarrow T[3] \rightarrow (3904433, "A")$`

Figure: Hashtable T

0		
1		
2	← 312692	B
3	← 3904433	A
4	← 5148949	C

Associative Arrays

The Hash Map

Searching:

- ▶ `search(3904433): $h(3904433) = 3 \Rightarrow T[3] \rightarrow (3904433, "A")$`
- ▶ `search(123459): $h(123459) = 4 \Rightarrow T[4]$`
 \Rightarrow Value with key 123459 does not exist

Figure: Hashtable T

0	
1	
2	← 312692 B
3	← 3904433 A
4	← 5148949 C

Associative Arrays

The Hash Map

Searching:

- ▶ `search(3904433): $h(3904433) = 3 \Rightarrow T[3] \rightarrow (3904433, "A")$`
- ▶ `search(123459): $h(123459) = 4 \Rightarrow T[4]$`
 \Rightarrow Value with key 123459 does not exist
- ▶ Search time for this example: $\mathcal{O}(1)$

Figure: Hashtable T

0	
1	
2	← 312692 B
3	← 3904433 A
4	← 5148949 C

Associative Arrays

Hash Collisions

Further inserting:

- ▶ `insert(876543, "D")`: $h(876543) = 3$

Figure: Hashtable T

0	
1	
2	← 312692 B
3	← 3904433 A
4	← 5148949 C

Associative Arrays

Hash Collisions

Further inserting:

- ▶ `insert(876543, "D")`: $h(876543) = 3$
 $\Rightarrow T[3] = (876543, "D") \Rightarrow$ Collision

Figure: Hashtable T

0	
1	
2	← 312692 B
3	← 3904433 A
4	← 5148949 C

Associative Arrays

Hash Collisions

Further inserting:

- ▶ `insert(876543, "D")`: $h(876543) = 3$
 $\Rightarrow T[3] = (876543, "D") \Rightarrow$ Collision
- ▶ This happens more often than expected
 - ▶ **Birthday problem:** With 23 people we have the probability of 50 % that 2 of them have birthday at the same day

Figure: Hashtable T

0	
1	
2	← 312692 B
3	← 3904433 A
4	← 5148949 C

Associative Arrays

Hash Collisions

Problem:

- ▶ Two keys are equal $h(x) = h(y)$ but not the values $x \neq y$

Associative Arrays

Hash Collisions

Problem:

- ▶ Two keys are equal $h(x) = h(y)$ but not the values $x \neq y$

Easiest Solution:

Associative Arrays

Hash Collisions

Problem:

- ▶ Two keys are equal $h(x) = h(y)$ but not the values $x \neq y$

Easiest Solution:

- ▶ Represent each bucket as list of key value pairs

Figure: Hashtable T

0	
1	
2	← 312692 B
3	← 3904433 A
4	← 5148949 C

Associative Arrays

Hash Collisions

Problem:

- ▶ Two keys are equal $h(x) = h(y)$ but not the values $x \neq y$

Easiest Solution:

- ▶ Represent each bucket as list of key value pairs
- ▶ Append new values to the end of the list

Figure: Hashtable T



Associative Arrays

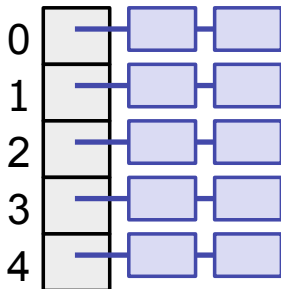
Expected Runtime

Best case:

- ▶ We have n keys which are equally distributed over m buckets
- ▶ We have $\approx \frac{n}{m}$ pairs per bucket
- ▶ The runtime for searching is nearly $\mathcal{O}(1)$ when **not** $n \gg m$

Best case

$(m = 5, n = 10)$



Associative Arrays

Expected Runtime

Worst case:

- ▶ All n keys are mapped onto the same bucket
- ▶ The runtime is $\Theta(n)$ for searching



Structure

Associative Arrays

- Introduction

- Hash Map

Universal Hashing

- Introduction

- Probability Calculation

- Proof

- Examples

Universal Hashing

Thought Experiment

Thought Experiment:

- ▶ A hash function is defined for a given key set

Universal Hashing

Thought Experiment

Thought Experiment:

- ▶ A hash function is defined for a given key set
- ▶ Find a set of keys resulting in a degenerated hash table

Universal Hashing

Thought Experiment

Thought Experiment:

- ▶ A *hash function* is defined for a given *key set*
- ▶ Find a *set of keys* resulting in a degenerated *hash table*
 - ▶ *The hash function stays fixed*

Universal Hashing

Thought Experiment

Thought Experiment:

- ▶ A **hash function** is defined for a given **key set**
- ▶ Find a **set of keys** resulting in a degenerated **hash table**
 - ▶ *The **hash function** stays fixed*
 - ▶ *For table size of 100: Try $100 \times (99 + 1)$ different numbers*

Universal Hashing

Thought Experiment

Thought Experiment:

- ▶ A **hash function** is defined for a given **key set**
- ▶ Find a **set of keys** resulting in a degenerated **hash table**
 - ▶ *The **hash function** stays fixed*
 - ▶ *For table size of 100: Try $100 \times (99 + 1)$ different numbers*
 - ▶ *Worst case: All 100 **key sets** map to one bucket*
- ▶ **Now:** Find a solution to avoid that problem

Universal Hashing

Idea

Solution: universal hashing

- Out of a set of hash functions we randomly choose one



Figure: Hash func. 1



Figure: Hash func. 2



Figure: Hash func. coll.

Universal Hashing

Idea

Solution: universal hashing

- ▶ Out of a set of hash functions we randomly choose one
- ▶ The **expected result** of the hash function is an equal distribution over the buckets



Figure: Hash func. 1



Figure: Hash func. 2



Figure: Hash func. coll.

Universal Hashing

Idea

Solution: universal hashing

- ▶ Out of a set of hash functions we randomly choose one
- ▶ The **expected result** of the hash function is an equal distribution over the buckets
- ▶ This hash function stays fixed for the lifetime of table
Optional: copy table with new hash when degenerated



Figure: Hash func. 1



Figure: Hash func. 2

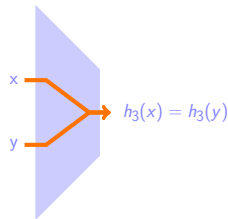


Figure: Hash func. coll.

Universal Hashing

Definition

Definition:

- ▶ We call \mathcal{U} the set (universum) of possible keys

Key universe \mathcal{U}



Universal Hashing

Definition

Definition:

- ▶ We call \mathcal{U} the set (universum) of possible keys
- ▶ The size m of the hash table T



Universal Hashing

Definition

Definition:

- ▶ We call \mathbb{U} the set (universum) of possible keys
- ▶ The size m of the hash table T
- ▶ Set of hash functions $\mathbb{H} = \{h_1, h_2, \dots, h_n\}$ with $h_i : \mathbb{U} \rightarrow \{0, \dots, m-1\}$



Figure: Hash function h_1

Universal Hashing

Definition

Definition:

- ▶ We call \mathbb{U} the set (universum) of possible keys
- ▶ The size m of the hash table T
- ▶ Set of hash functions $\mathbb{H} = \{h_1, h_2, \dots, h_n\}$ with $h_i : \mathbb{U} \rightarrow \{0, \dots, m-1\}$



Figure: Hash function h_1

Universal Hashing

Definition

Definition:

- ▶ We call \mathbb{U} the set (universum) of possible keys
- ▶ The size m of the hash table T
- ▶ Set of hash functions $\mathbb{H} = \{h_1, h_2, \dots, h_n\}$ with $h_i : \mathbb{U} \rightarrow \{0, \dots, m-1\}$
- ▶ Idea: runtime should be $O(1 + \frac{|\mathbb{S}|}{m})$, where $\frac{|\mathbb{S}|}{m}$ is the table load



Figure: Hash function h_1

Universal Hashing

Definition

- We choose two random keys
 $x, y \in \mathbb{U} \mid x \neq y$



Figure: Set of hash functions \mathbb{H}

Universal Hashing

Definition

- ▶ We choose two random keys $x, y \in \mathbb{U} \mid x \neq y$
- ▶ An average of 3 out of 15 functions produce collisions



Figure: Set of hash functions \mathbb{H}

Universal Hashing

Definition

Definition: \mathbb{H} is c -universal if $\forall x, y \in \mathbb{U} \mid x \neq y :$

Number of hash functions that create collisions

$$\frac{|\{h \in \mathbb{H} : h(x) = h(y)\}|}{|\mathbb{H}|} \leq c \cdot \frac{1}{m}, \quad c \in \mathbb{R}$$

Number of hash functions

Universal Hashing

Definition

Definition: \mathbb{H} is c -universal if $\forall x, y \in \mathbb{U} \mid x \neq y :$

Number of hash functions that create collisions

$$\frac{|\{h \in \mathbb{H} : h(x) = h(y)\}|}{|\mathbb{H}|} \leq c \cdot \frac{1}{m}, \quad c \in \mathbb{R}$$

Number of hash functions

- ▶ With other words, given a arbitrary but fixed pair x, y .
If $h \in \mathbb{H}$ is chosen randomly then

Universal Hashing

Definition

Definition: \mathbb{H} is c -universal if $\forall x, y \in \mathbb{U} \mid x \neq y :$

Number of hash functions that create collisions

$$\frac{|\{h \in \mathbb{H} : h(x) = h(y)\}|}{|\mathbb{H}|} \leq c \cdot \frac{1}{m}, \quad c \in \mathbb{R}$$

Number of hash functions

- ▶ With other words, given a arbitrary but fixed pair x, y .
If $h \in \mathbb{H}$ is chosen randomly then

$$\text{Prob}(h(x) = h(y)) \leq c \cdot \frac{1}{m}$$

Universal Hashing

Definition

Definition: \mathbb{H} is c -universal if $\forall x, y \in \mathbb{U} \mid x \neq y :$

Number of hash functions that create collisions

$$\frac{|\{h \in \mathbb{H} : h(x) = h(y)\}|}{|\mathbb{H}|} \leq c \cdot \frac{1}{m}, \quad c \in \mathbb{R}$$

Number of hash functions

- With other words, given a arbitrary but fixed pair x, y .
If $h \in \mathbb{H}$ is chosen randomly then

$$\text{Prob}(h(x) = h(y)) \leq c \cdot \frac{1}{m}$$

Note: If the hash function assigns each key x and y randomly to buckets then:

Universal Hashing

Definition

Definition: \mathbb{H} is c -universal if $\forall x, y \in \mathbb{U} \mid x \neq y :$

Number of hash functions that create collisions

$$\frac{|\{h \in \mathbb{H} : h(x) = h(y)\}|}{|\mathbb{H}|} \leq c \cdot \frac{1}{m}, \quad c \in \mathbb{R}$$

Number of hash functions

- ▶ With other words, given a arbitrary but fixed pair x, y .
If $h \in \mathbb{H}$ is chosen randomly then

$$\text{Prob}(h(x) = h(y)) \leq c \cdot \frac{1}{m}$$

Note: If the hash function assigns each key x and y randomly to buckets then:

$$\text{Prob}(\text{Collision}) = \frac{1}{m} \Leftrightarrow c = 1$$

Universal Hashing

Definition

- ▶ \mathcal{U} : Key universe
- ▶ \mathcal{S} : Used Keys
- ▶ $\mathcal{S}_i \subseteq \mathcal{S}$: Keys mapping to Bucket i (“synonyms”)
- ▶ Ideal would be $|\mathcal{S}_i| = \frac{|\mathcal{S}|}{m}$



Figure: Hash function $h \in \mathbb{H}$

Universal Hashing

Definition

- ▶ Let \mathcal{H} be a c -universal class of hash functions

Universal Hashing

Definition

- ▶ Let \mathcal{H} be a c -universal class of hash functions
- ▶ Let \mathcal{S} be a set of keys and $h \in \mathcal{H}$ selected randomly

Universal Hashing

Definition

- ▶ Let \mathbb{H} be a c -universal class of hash functions
- ▶ Let \mathbb{S} be a set of keys and $h \in \mathbb{H}$ selected randomly
- ▶ Let \mathbb{S}_i be the key x for which $h(x) = i$

Universal Hashing

Definition

- ▶ Let \mathbb{H} be a c -universal class of hash functions
- ▶ Let \mathbb{S} be a set of keys and $h \in \mathbb{H}$ selected randomly
- ▶ Let \mathbb{S}_i be the key x for which $h(x) = i$
- ▶ The expected average number of elements to search through per bucket is

$$\mathbb{E}[|\mathbb{S}_i|] \leq 1 + c \cdot \frac{|\mathbb{S}|}{m}$$

Universal Hashing

Definition

- ▶ Let \mathbb{H} be a c -universal class of hash functions
- ▶ Let \mathbb{S} be a set of keys and $h \in \mathbb{H}$ selected randomly
- ▶ Let \mathbb{S}_i be the key x for which $h(x) = i$
- ▶ The expected average number of elements to search through per bucket is

$$\mathbb{E}[|\mathbb{S}_i|] \leq 1 + c \cdot \frac{|\mathbb{S}|}{m}$$

- ▶ Particular: If $(m = \Omega(|\mathbb{S}|))$ then $\mathbb{E}[|\mathbb{S}_i|] = \mathcal{O}(1)$

Structure

Associative Arrays

Introduction

Hash Map

Universal Hashing

Introduction

Probability Calculation

Proof

Examples

Universal Hashing

Probability Calculation

Universal Hashing

Probability Calculation

- ▶ We just discuss the discrete case

Universal Hashing

Probability Calculation

- ▶ We just discuss the discrete case
- ▶ Probability space Ω with elementary (simple) events

Universal Hashing

Probability Calculation

- ▶ We just discuss the discrete case
- ▶ Probability space Ω with elementary (simple) events
- ▶ Events e have probabilities ...

$$\sum_{e \in \Omega} P(e) = 1$$

Universal Hashing

Probability Calculation

- ▶ We just discuss the discrete case
- ▶ Probability space Ω with elementary (simple) events
- ▶ Events e have probabilities ...

$$\sum_{e \in \Omega} P(e) = 1$$

- ▶ The probability for a subset of events $E \subseteq \Omega$ is

$$P(E) = \sum_{e \in E} P(e) \mid e \in E$$

Universal Hashing

Probability Calculation

- ▶ We just discuss the discrete case
- ▶ Probability space Ω with elementary (simple) events
- ▶ Events e have probabilities ...

$$\sum_{e \in \Omega} P(e) = 1$$

- ▶ The probability for a subset of events $E \subseteq \Omega$ is

$$P(E) = \sum_{e \in E} P(e)$$

Table: Throwing a dice

e	$P(e)$
1	1/6
2	1/6
3	1/6
4	1/6
5	1/6
6	1/6

Universal Hashing

Probability Calculation

Example:

Universal Hashing

Probability Calculation

Example:

- ▶ Rolling a dice twice ($\Omega = \{1, \dots, 6\}^2$)

Universal Hashing

Probability Calculation

Example:

- ▶ Rolling a dice twice ($\Omega = \{1, \dots, 6\}^2$)
- ▶ Each event $e \in \Omega$ has the probability $P(e) = 1/36$

Table: Throwing a dice twice

e	$P(e)$
(1, 1)	1/36
(1, 2)	1/36
(1, 3)	1/36
...	...
(6, 5)	1/36
(6, 6)	1/36

Universal Hashing

Probability Calculation

Example:

- ▶ Rolling a dice twice ($\Omega = \{1, \dots, 6\}^2$)
- ▶ Each event $e \in \Omega$ has the probability
 $P(e) = 1/36$
- ▶ $E =$ if both results are even, then
 $P(E) =$

Table: Throwing a dice twice

e	$P(e)$
(1, 1)	1/36
(1, 2)	1/36
(1, 3)	1/36
...	...
(6, 5)	1/36
(6, 6)	1/36

Universal Hashing

Probability Calculation

Example:

- ▶ Random variable

Universal Hashing

Probability Calculation

Example:

- ▶ Random variable
 - ▶ Assigns a number to the result of an experiment

Universal Hashing

Probability Calculation

Example:

- ▶ Random variable
 - ▶ Assigns a number to the result of an experiment
 - ▶ For example: X = Sum of results for rolling twice

Universal Hashing

Probability Calculation

Example:

- ▶ Random variable
 - ▶ Assigns a number to the result of an experiment
 - ▶ For example: $X = \text{Sum of results for rolling twice}$
 - ▶ $X = 12$ and $X \geq 7$ are regarded as events

Table: Throwing a dice twice

e	$P(e)$	X
(1, 1)	$1/36$	2
(1, 2)	$1/36$	3
(1, 3)	$1/36$	4
...
(6, 5)	$1/36$	11
(6, 6)	$1/36$	12

Universal Hashing

Probability Calculation

Example:

- ▶ Random variable
 - ▶ Assigns a number to the result of an experiment
 - ▶ For example: X = Sum of results for rolling twice
 - ▶ $X = 12$ and $X \geq 7$ are regarded as events
 - ▶ Example 1: $P(X = 2) =$

Table: Throwing a dice twice

e	$P(e)$	X
(1, 1)	$1/36$	2
(1, 2)	$1/36$	3
(1, 3)	$1/36$	4
...
(6, 5)	$1/36$	11
(6, 6)	$1/36$	12

Universal Hashing

Probability Calculation

Example:

- ▶ Random variable
 - ▶ Assigns a number to the result of an experiment
 - ▶ For example: X = Sum of results for rolling twice
 - ▶ $X = 12$ and $X \geq 7$ are regarded as events
 - ▶ Example 1: $P(X = 2) =$
 - ▶ Example 2: $P(X = 4) =$

Table: Throwing a dice twice

e	$P(e)$	X
(1, 1)	$1/36$	2
(1, 2)	$1/36$	3
(1, 3)	$1/36$	4
...
(6, 5)	$1/36$	11
(6, 6)	$1/36$	12

Universal Hashing

Probability Calculation

Expected value is defined as $\mathbb{E}(X) = \sum (k \cdot P(X = k))$

Universal Hashing

Probability Calculation

Expected value is defined as $\mathbb{E}(X) = \sum (k \cdot P(X = k))$

- Intuitive: The weighted average of possible values of X , where the weights are the probabilities of the values

Universal Hashing

Probability Calculation

Expected value is defined as $\mathbb{E}(X) = \sum (k \cdot P(X = k))$

- Intuitive: The weighted average of possible values of X , where the weights are the probabilities of the values

Table: Throwing a dice once

X	$P(X)$
1	$1/6$
2	$1/6$
3	$1/6$
4	$1/6$
5	$1/6$
6	$1/6$

Table: Throwing a dice twice

X	$P(X)$
2	$1/36$
3	$2/36$
4	$3/36$
...	...
11	$2/36$
12	$1/36$

Universal Hashing

Probability Calculation

Expected value is defined as $\mathbb{E}(X) = \sum (k \cdot P(X = k))$

- Intuitive: The weighted average of possible values of X , where the weights are the probabilities of the values

Table: Throwing a dice once

X	$P(X)$
1	$1/6$
2	$1/6$
3	$1/6$
4	$1/6$
5	$1/6$
6	$1/6$

Table: Throwing a dice twice

X	$P(X)$
2	$1/36$
3	$2/36$
4	$3/36$
...	...
11	$2/36$
12	$1/36$

- Example rolling once:

Universal Hashing

Probability Calculation

Expected value is defined as $\mathbb{E}(X) = \sum (k \cdot P(X = k))$

- Intuitive: The weighted average of possible values of X , where the weights are the probabilities of the values

Table: Throwing a dice once

X	$P(X)$
1	$1/6$
2	$1/6$
3	$1/6$
4	$1/6$
5	$1/6$
6	$1/6$

Table: Throwing a dice twice

X	$P(X)$
2	$1/36$
3	$2/36$
4	$3/36$
...	...
11	$2/36$
12	$1/36$

- Example rolling once: $\mathbb{E}(X) = 1 \cdot \frac{1}{6} + 2 \cdot \frac{1}{6} + \dots + 6 \cdot \frac{1}{6} = 3.5$

Universal Hashing

Probability Calculation

Expected value is defined as $\mathbb{E}(X) = \sum (k \cdot P(X = k))$

- Intuitive: The weighted average of possible values of X , where the weights are the probabilities of the values

Table: Throwing a dice once

X	$P(X)$
1	$1/6$
2	$1/6$
3	$1/6$
4	$1/6$
5	$1/6$
6	$1/6$

Table: Throwing a dice twice

X	$P(X)$
2	$1/36$
3	$2/36$
4	$3/36$
...	...
11	$2/36$
12	$1/36$

- Example rolling once: $\mathbb{E}(X) = 1 \cdot \frac{1}{6} + 2 \cdot \frac{1}{6} + \dots + 6 \cdot \frac{1}{6} = 3.5$
- Example rolling twice:

Universal Hashing

Probability Calculation

Expected value is defined as $\mathbb{E}(X) = \sum (k \cdot P(X = k))$

- Intuitive: The weighted average of possible values of X , where the weights are the probabilities of the values

Table: Throwing a dice once

X	$P(X)$
1	$1/6$
2	$1/6$
3	$1/6$
4	$1/6$
5	$1/6$
6	$1/6$

Table: Throwing a dice twice

X	$P(X)$
2	$1/36$
3	$2/36$
4	$3/36$
...	...
11	$2/36$
12	$1/36$

- Example rolling once: $\mathbb{E}(X) = 1 \cdot \frac{1}{6} + 2 \cdot \frac{1}{6} + \dots + 6 \cdot \frac{1}{6} = 3.5$
- Example rolling twice: $\mathbb{E}(X) = 2 \cdot \frac{1}{36} + 3 \cdot \frac{2}{36} + \dots + 12 \cdot \frac{1}{36} = 7$

Universal Hashing

Probability Calculation

Sum of expected values: For arbitrary discrete random variables X_1, \dots, X_n we can write:

$$\mathbb{E}(X_1 + \dots + X_n) = \mathbb{E}(X_1) + \dots + \mathbb{E}(X_n)$$

Universal Hashing

Probability Calculation

Sum of expected values: For arbitrary discrete random variables X_1, \dots, X_n we can write:

$$\mathbb{E}(X_1 + \dots + X_n) = \mathbb{E}(X_1) + \dots + \mathbb{E}(X_n)$$

Example: Throwing two dice

Universal Hashing

Probability Calculation

Sum of expected values: For arbitrary discrete random variables X_1, \dots, X_n we can write:

$$\mathbb{E}(X_1 + \dots + X_n) = \mathbb{E}(X_1) + \dots + \mathbb{E}(X_n)$$

Example: Throwing two dice

- ▶ X_1 : Expected result of dice 1: $\mathbb{E}(X_1) = 3.5$

Universal Hashing

Probability Calculation

Sum of expected values: For arbitrary discrete random variables X_1, \dots, X_n we can write:

$$\mathbb{E}(X_1 + \dots + X_n) = \mathbb{E}(X_1) + \dots + \mathbb{E}(X_n)$$

Example: Throwing two dice

- ▶ X_1 : Expected result of dice 1: $\mathbb{E}(X_1) = 3.5$
- ▶ X_2 : Expected result of dice 2: $\mathbb{E}(X_2) = 3.5$

Universal Hashing

Probability Calculation

Sum of expected values: For arbitrary discrete random variables X_1, \dots, X_n we can write:

$$\mathbb{E}(X_1 + \dots + X_n) = \mathbb{E}(X_1) + \dots + \mathbb{E}(X_n)$$

Example: Throwing two dice

- ▶ X_1 : Expected result of dice 1: $\mathbb{E}(X_1) = 3.5$
- ▶ X_2 : Expected result of dice 2: $\mathbb{E}(X_2) = 3.5$
- ▶ $X = X_1 + X_2$: Expected total number:

$$\mathbb{E}(X) = \mathbb{E}(X_1 + X_2) = \mathbb{E}(X_1) + \mathbb{E}(X_2) = 3.5 + 3.5 = 7$$

Universal Hashing

Probability Calculation

Corollary:

The probability of the event E is $p = P(E)$. Let X be the occurrences of the event E and n be the number of executions of the experiment. Then $\mathbb{E}(X) = n \cdot P(E) = n \cdot p$

Universal Hashing

Probability Calculation

Corollary:

The probability of the event E is $p = P(E)$. Let X be the occurrences of the event E and n be the number of executions of the experiment. Then $\mathbb{E}(X) = n \cdot P(E) = n \cdot p$

Example (Rolling the dice 60 times:)

$$\mathbb{E}(\text{occurrences of } 6) = \frac{1}{6} \cdot 60 = 10$$

Universal Hashing

Probability Calculation

Proof Corollary:

Indicator variable: X_i

Universal Hashing

Probability Calculation

Proof Corollary:

Indicator variable: X_i

$$X_i = \begin{cases} 1, & \text{if event occurs} \\ 0, & \text{else} \end{cases}$$

$$\Rightarrow X = \sum_{i=1}^n X_i$$

Universal Hashing

Probability Calculation

Proof Corollary:

Indicator variable: X_i

$$X_i = \begin{cases} 1, & \text{if event occurs} \\ 0, & \text{else} \end{cases}$$

$$\Rightarrow X = \sum_{i=1}^n X_i$$

$$\mathbb{E}(X) = \mathbb{E}\left(\sum_{i=1}^n X_i\right) = \sum_{i=1}^n \mathbb{E}(X_i) \stackrel{\text{def. } \mathbb{E}\text{-value}}{=} \sum_{i=1}^n p = n \cdot p$$



Universal Hashing

Probability Calculation

Proof Corollary:

Indicator variable: X_i

$$X_i = \begin{cases} 1, & \text{if event occurs} \\ 0, & \text{else} \end{cases}$$

$$\Rightarrow X = \sum_{i=1}^n X_i$$

$$\mathbb{E}(X) = \mathbb{E}\left(\sum_{i=1}^n X_i\right) = \sum_{i=1}^n \mathbb{E}(X_i) \stackrel{\text{def. } \mathbb{E}\text{-value}}{=} \sum_{i=1}^n p = n \cdot p$$

□

Def. \mathbb{E} -value: $\mathbb{E}(X_i) = 0 \cdot P(X_i = 0) + 1 \cdot P(X_i = 1) = P(X_i = 1)$

Structure

Associative Arrays

Introduction

Hash Map

Universal Hashing

Introduction

Probability Calculation

Proof

Examples

Universal Hashing

Proof

Given:

- ▶ We pick two random keys $x, y \in \mathbb{S} \mid x \neq y$ and a random hash function $h \in \mathbb{H}$

Universal Hashing

Proof

Given:

- ▶ We pick two random keys $x, y \in \mathbb{S} \mid x \neq y$ and a random hash function $h \in \mathbb{H}$
- ▶ We know the probability of a collision:

$$P(h(x) = h(y)) \leq c \cdot \frac{1}{m}$$

Universal Hashing

Proof

Given:

- ▶ We pick two random keys $x, y \in \mathbb{S} \mid x \neq y$ and a random hash function $h \in \mathbb{H}$
- ▶ We know the probability of a collision:

$$P(h(x) = h(y)) \leq c \cdot \frac{1}{m}$$

To proof:

$$\mathbb{E}[|S_i|] \leq 1 + c \cdot \frac{|\mathbb{S}|}{m} \quad \forall i$$

Universal Hashing

Proof

We know:

$$\mathbb{S}_i = \{x \in \mathbb{S} : h(x) = i\}$$

Universal Hashing

Proof

We know:

$$\mathbb{S}_i = \{x \in \mathbb{S} : h(x) = i\}$$

If $\mathbb{S}_i = \emptyset \Rightarrow |\mathbb{S}_i| = 0$

Universal Hashing

Proof

We know:

$$\mathbb{S}_i = \{x \in \mathbb{S} : h(x) = i\}$$

If $\mathbb{S}_i = \emptyset \Rightarrow |\mathbb{S}_i| = 0$ otherwise, let $x \in \mathbb{S}_i$ be any key

Universal Hashing

Proof

We know:

$$\mathbb{S}_i = \{x \in \mathbb{S} : h(x) = i\}$$

If $\mathbb{S}_i = \emptyset \Rightarrow |\mathbb{S}_i| = 0$ otherwise, let $x \in \mathbb{S}_i$ be any key

We define an indicator variable:

$$I_y = \begin{cases} 1, & \text{if } h(y) = i \\ 0, & \text{else} \end{cases} \quad y \in \mathbb{S} \setminus \{x\}$$

Universal Hashing

Proof

We know:

$$\mathbb{S}_i = \{x \in \mathbb{S} : h(x) = i\}$$

If $\mathbb{S}_i = \emptyset \Rightarrow |\mathbb{S}_i| = 0$ otherwise, let $x \in \mathbb{S}_i$ be any key

We define an indicator variable:

$$I_y = \begin{cases} 1, & \text{if } h(y) = i \\ 0, & \text{else} \end{cases} \quad y \in \mathbb{S} \setminus \{x\}$$

$$\Rightarrow |\mathbb{S}_i| = 1 + \sum_{y \in \mathbb{S} \setminus x} I_y$$

Universal Hashing

Proof

We know:

$$\mathbb{S}_i = \{x \in \mathbb{S} : h(x) = i\}$$

If $\mathbb{S}_i = \emptyset \Rightarrow |\mathbb{S}_i| = 0$ otherwise, let $x \in \mathbb{S}_i$ be any key

We define an indicator variable:

$$I_y = \begin{cases} 1, & \text{if } h(y) = i \\ 0, & \text{else} \end{cases} \quad y \in \mathbb{S} \setminus \{x\}$$

$$\Rightarrow |\mathbb{S}_i| = 1 + \sum_{y \in \mathbb{S} \setminus x} I_y$$

$$\Rightarrow \mathbb{E}(|\mathbb{S}_i|) = \mathbb{E}\left(1 + \sum_{y \in \mathbb{S} \setminus x} I_y\right) = 1 + \sum_{y \in \mathbb{S} \setminus x} \mathbb{E}(I_y)$$

Universal Hashing

Proof

Auxiliary calculation:

$$\begin{aligned}\mathbb{E}[I_y] &= P(I_y = 1) \\ &= P(h(y) = i) \\ &= P(h(y) = h(x)) \\ &\leq c \cdot \frac{1}{m}\end{aligned}$$

Universal Hashing

Proof

Auxiliary calculation:

$$\begin{aligned}\mathbb{E}[I_y] &= P(I_y = 1) \\ &= P(h(y) = i) \\ &= P(h(y) = h(x)) \\ &\leq c \cdot \frac{1}{m}\end{aligned}$$

Universal Hashing

Proof

Auxiliary calculation:

$$\begin{aligned}\mathbb{E}[I_y] &= P(I_y = 1) \\ &= P(h(y) = i) \\ &= P(h(y) = h(x)) \\ &\leq c \cdot \frac{1}{m}\end{aligned}$$

Hence:

$$\begin{aligned}\mathbb{E}[|S_i|] &= 1 + \sum_{y \in S \setminus x} \mathbb{E}[I_y] \leq 1 + \sum_{y \in S \setminus x} c \cdot \frac{1}{m} \\ &= 1 + (|S| - 1) \cdot c \cdot \frac{1}{m} \\ &\leq 1 + |S| \cdot c \cdot \frac{1}{m} \\ &= 1 + c \cdot \frac{|S|}{m}\end{aligned}$$

□

Universal Hashing

Proof

Auxiliary calculation:

$$\begin{aligned}\mathbb{E}[I_y] &= P(I_y = 1) \\ &= P(h(y) = i) \\ &= P(h(y) = h(x)) \\ &\leq c \cdot \frac{1}{m}\end{aligned}$$

Hence: $\mathbb{E}[|S_i|] = 1 + \sum_{y \in S \setminus x} \mathbb{E}[I_y] \leq 1 + \sum_{y \in S \setminus x} c \cdot \frac{1}{m}$

$$\leq 1 + |S| \cdot c \cdot \frac{1}{m}$$

$$= 1 + c \cdot \frac{|S|}{m}$$

□

Universal Hashing

Proof

Auxiliary calculation:

$$\begin{aligned}\mathbb{E}[I_y] &= P(I_y = 1) \\ &= P(h(y) = i) \\ &= P(h(y) = h(x)) \\ &\leq c \cdot \frac{1}{m}\end{aligned}$$

$$\begin{aligned}\textbf{Hence: } \mathbb{E}[|S_i|] &= 1 + \sum_{y \in S \setminus x} \mathbb{E}[I_y] \leq 1 + \sum_{y \in S \setminus x} c \cdot \frac{1}{m} \\ &= 1 + (|S| - 1) \cdot c \cdot \frac{1}{m} \\ &\leq 1 + |S| \cdot c \cdot \frac{1}{m} \\ &= 1 + c \cdot \frac{|S|}{m}\end{aligned}$$

□

Structure

Associative Arrays

Introduction

Hash Map

Universal Hashing

Introduction

Probability Calculation

Proof

Examples

Universal Hashing

Examples

Negative example:

Universal Hashing

Examples

Negative example:

- ▶ The set of all h for which $h_a(x) = (a \cdot x) \bmod m$, for a $a \in \mathbb{U}$

Universal Hashing

Examples

Negative example:

- ▶ The set of all h for which $h_a(x) = (a \cdot x) \bmod m$, for a $a \in \mathbb{U}$
- ▶ Is not c -universal. Why?

Universal Hashing

Examples

Negative example:

- ▶ The set of all h for which $h_a(x) = (a \cdot x) \bmod m$, for a $a \in \mathbb{U}$
- ▶ Is not c -universal. Why?
- ▶ If universal:

$$\forall x, y \quad x \neq y: \frac{|\{h \in \mathbb{H} : h(x) = h(y)\}|}{|\mathbb{H}|} \leq c \cdot \frac{1}{m}$$

Universal Hashing

Examples

Negative example:

- ▶ The set of all h for which $h_a(x) = (a \cdot x) \bmod m$, for a $a \in \mathbb{U}$
- ▶ Is not c -universal. Why?
- ▶ If universal:

$$\forall x, y \quad x \neq y: \frac{|\{h \in \mathbb{H} : h(x) = h(y)\}|}{|\mathbb{H}|} \leq c \cdot \frac{1}{m}$$

- ▶ Which x, y lead to a relative collision count bigger than $\frac{c}{m}$?

Universal Hashing

Examples

Positive example:

- ▶ Let p be a big prime number, $p > m$ and $p \geq |\mathbb{U}|$

Universal Hashing

Examples

Positive example:

- ▶ Let p be a big prime number, $p > m$ and $p \geq |\mathbb{U}|$
- ▶ Let \mathbb{H} be the set of all h for which:

$$h_{a,b}(x) = ((a \cdot x + b) \bmod p) \bmod m,$$

where $1 \leq a < p$, $0 \leq b < p$

Universal Hashing

Examples

Positive example:

- ▶ Let p be a big prime number, $p > m$ and $p \geq |\mathbb{U}|$
- ▶ Let \mathbb{H} be the set of all h for which:

$$h_{a,b}(x) = ((a \cdot x + b) \bmod p) \bmod m,$$

where $1 \leq a < p$, $0 \leq b < p$

- ▶ This is \approx 1-universal, see Exercise 4.11 in Mehlhorn/Sanders

Universal Hashing

Examples

Positive example:

- ▶ Let p be a big prime number, $p > m$ and $p \geq |\mathbb{U}|$
- ▶ Let \mathbb{H} be the set of all h for which:

$$h_{a,b}(x) = ((a \cdot x + b) \bmod p) \bmod m,$$

where $1 \leq a < p$, $0 \leq b < p$

- ▶ This is ≈ 1 -universal, see Exercise 4.11 in Mehlhorn/Sanders
- ▶ E.g.: $U = \{0, \dots, 99\}$, $p = 101$, $a = 47$, $b = 5$

Universal Hashing

Examples

Positive example:

- ▶ Let p be a big prime number, $p > m$ and $p \geq |\mathbb{U}|$
- ▶ Let \mathbb{H} be the set of all h for which:

$$h_{a,b}(x) = ((a \cdot x + b) \bmod p) \bmod m,$$

where $1 \leq a < p$, $0 \leq b < p$

- ▶ This is ≈ 1 -universal, see Exercise 4.11 in Mehlhorn/Sanders
- ▶ E.g.: $U = \{0, \dots, 99\}$, $p = 101$, $a = 47$, $b = 5$
- ▶ Then $h(x) = ((47 \cdot x + 5) \bmod 101) \bmod m$

Universal Hashing

Examples

Positive example:

- ▶ Let p be a big prime number, $p > m$ and $p \geq |\mathbb{U}|$
- ▶ Let \mathbb{H} be the set of all h for which:

$$h_{a,b}(x) = ((a \cdot x + b) \bmod p) \bmod m,$$

where $1 \leq a < p$, $0 \leq b < p$

- ▶ This is \approx 1-universal, see Exercise 4.11 in Mehlhorn/Sanders
- ▶ E.g.: $U = \{0, \dots, 99\}$, $p = 101$, $a = 47$, $b = 5$
- ▶ Then $h(x) = ((47 \cdot x + 5) \bmod 101) \bmod m$
- ▶ Easy to implement but hard to proof

Universal Hashing

Examples

Positive example:

- ▶ Let p be a big prime number, $p > m$ and $p \geq |\mathbb{U}|$
- ▶ Let \mathbb{H} be the set of all h for which:

$$h_{a,b}(x) = ((a \cdot x + b) \bmod p) \bmod m,$$

where $1 \leq a < p$, $0 \leq b < p$

- ▶ This is ≈ 1 -universal, see Exercise 4.11 in Mehlhorn/Sanders
- ▶ E.g.: $U = \{0, \dots, 99\}$, $p = 101$, $a = 47$, $b = 5$
- ▶ Then $h(x) = ((47 \cdot x + 5) \bmod 101) \bmod m$
- ▶ Easy to implement but hard to proof
- ▶ Exercise: show empirically that it is 2-universal

Universal Hashing

Examples

Positive example:

- ▶ The set of hash functions is c -universal:

$$h_a(x) = a \bullet x \mod m, \quad a \in \mathbb{U}$$

Universal Hashing

Examples

Positive example:

- ▶ The set of hash functions is c -universal:

$$h_a(x) = a \bullet x \mod m, \quad a \in \mathbb{U}$$

- ▶ We define:

$$a = \sum_{0, \dots, k-1} a_i \cdot m^i, \quad k = \text{ceil}(\log_m |\mathbb{U}|)$$

$$x = \sum_{0, \dots, k-1} x_i \cdot m^i$$

Universal Hashing

Examples

Positive example:

- ▶ The set of hash functions is c -universal:

$$h_a(x) = a \bullet x \mod m, \quad a \in \mathbb{U}$$

- ▶ We define:

$$a = \sum_{0, \dots, k-1} a_i \cdot m^i, \quad k = \text{ceil}(\log_m |\mathbb{U}|)$$

$$x = \sum_{0, \dots, k-1} x_i \cdot m^i$$

- ▶ **Intuitive:** Scalar product with base m

$$a \bullet x = \sum_{0, \dots, k-1} a_i \cdot x_i$$

Universal Hashing

Examples

Example ($\mathbb{U} = \{0, \dots, 999\}$, $m = 10$, $a = 348$)

With $a = 348$: $a_2 = 3$, $a_1 = 4$, $a_0 = 8$

$$\begin{aligned}h_{348}(x) &= (a_2 \cdot x_2 + a_1 \cdot x_1 + a_0 \cdot x_0) \mod m \\&= (3x_2 + 4x_1 + 8x_0) \mod 10\end{aligned}$$

With $x = 127$: $x_2 = 1$, $x_1 = 2$, $x_0 = 7$

$$\begin{aligned}h_{348}(127) &= (3 \cdot x_2 + 4 \cdot x_1 + 8 \cdot x_0) \mod 10 \\&= (3 \cdot 1 + 4 \cdot 2 + 8 \cdot 7) \mod 10 \\&= 7\end{aligned}$$

Further Literature

► General for this Lecture

[CRL01] Thomas H. Cormen, Ronald L. Rivest, and Charles E. Leiserson.

Introduction to Algorithms.

MIT Press, Cambridge, Mass, 2001.

[MS08] Kurt Mehlhorn and Peter Sanders.

Algorithms and data structures, 2008.

<https://people.mpi-inf.mpg.de/~mehlhorn/ftp/Mehlhorn-Sanders-Toolbox.pdf>.

Further Literature

► Hash Map - Theory

[Wik] [Hash table](#)

https://en.wikipedia.org/wiki/Hash_table

► Hash Map - Implementations / API

[Cpp] [C++ - hash_map](#)

http://www.sgi.com/tech/stl/hash_map.html

[Jav] [Java - HashMap](#)

<https://docs.oracle.com/javase/7/docs/api/java/util/HashMap.html>

[Pyt] [Python - Dictionaries \(Hash table\)](#)

https://en.wikipedia.org/wiki/Hash_table