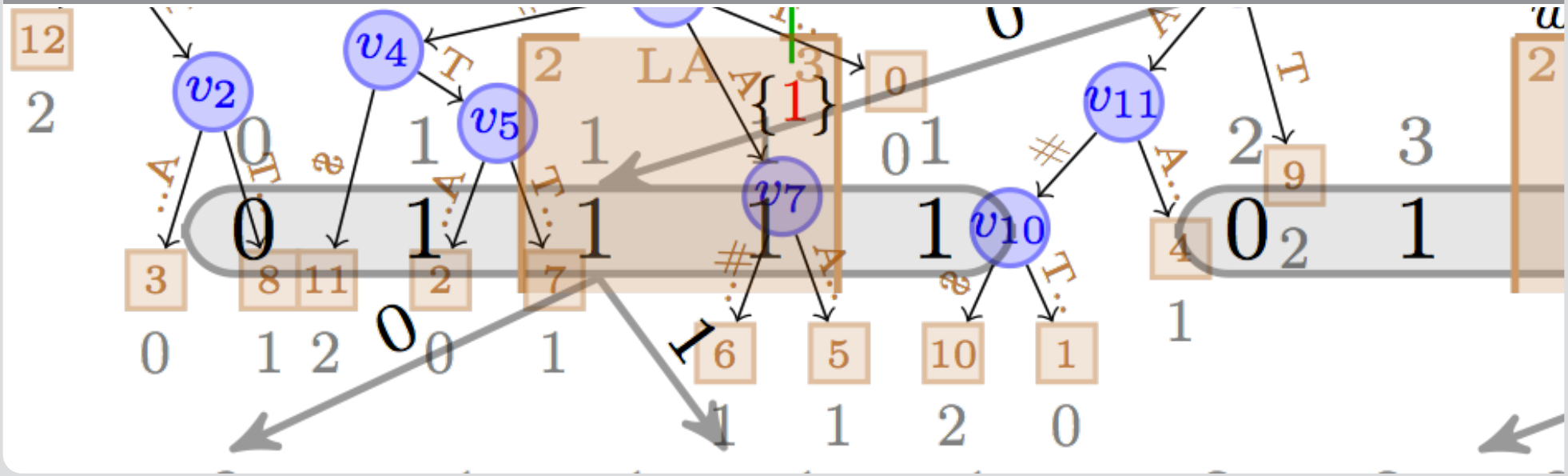
Karlsruhe Institute of Technology

Advanced Data Structures

Simon Gog – *gog@kit.edu*

Institute of Theoretical Informatics - Algorithmics



Mathematical preliminaries

- *Sample space* S is the set of outcomes of an experiment
- For $x \in S$, the *probability* $\Pr(x)$ of x is a number between 0 and 1, such that $\sum_{x \in S} \Pr(x) = 1$
- An *event* is a subset V of the sample space S
- The probability of V is $\Pr(V) = \sum_{x \in V} \Pr(x)$
- A *random variable* (r.v.) X is a function $S \to \mathbb{R}$
- The probability of Y taking value y is

$$\Pr(Y = y) = \sum_{\substack{x \in S \\ Y(x) = y}} \Pr(x)$$

- The *expected value* $\mathbb{E}(Y)$ of a r.v. Y is

$$\mathbb{E}(Y) = \sum_{x \in S} Y(x) \cdot \Pr(x)$$

Mathematical preliminaries

Union bound / Boole's inequality

For any sequence of events $V_0, V_1, \ldots, V_{k-1}$ it holds

$$\Pr(V_0 \cup V_1 \cup \cdots \cup V_{k-1}) \leq \sum_{i=0}^{k-1} \Pr(V_i)$$

Linearity of expectation

Let $Y_0, Y_2, \ldots, Y_{k-1}$ be k random variables. Then

$$\mathbb{E}(\sum_{i=0}^{k-1} Y_i) = \sum_{i=0}^{k-1} \mathbb{E}(Y_i)$$

Mathematical preliminaries

Markov's inequality

Let Y be a non-negative r.v. that only takes integer values, then

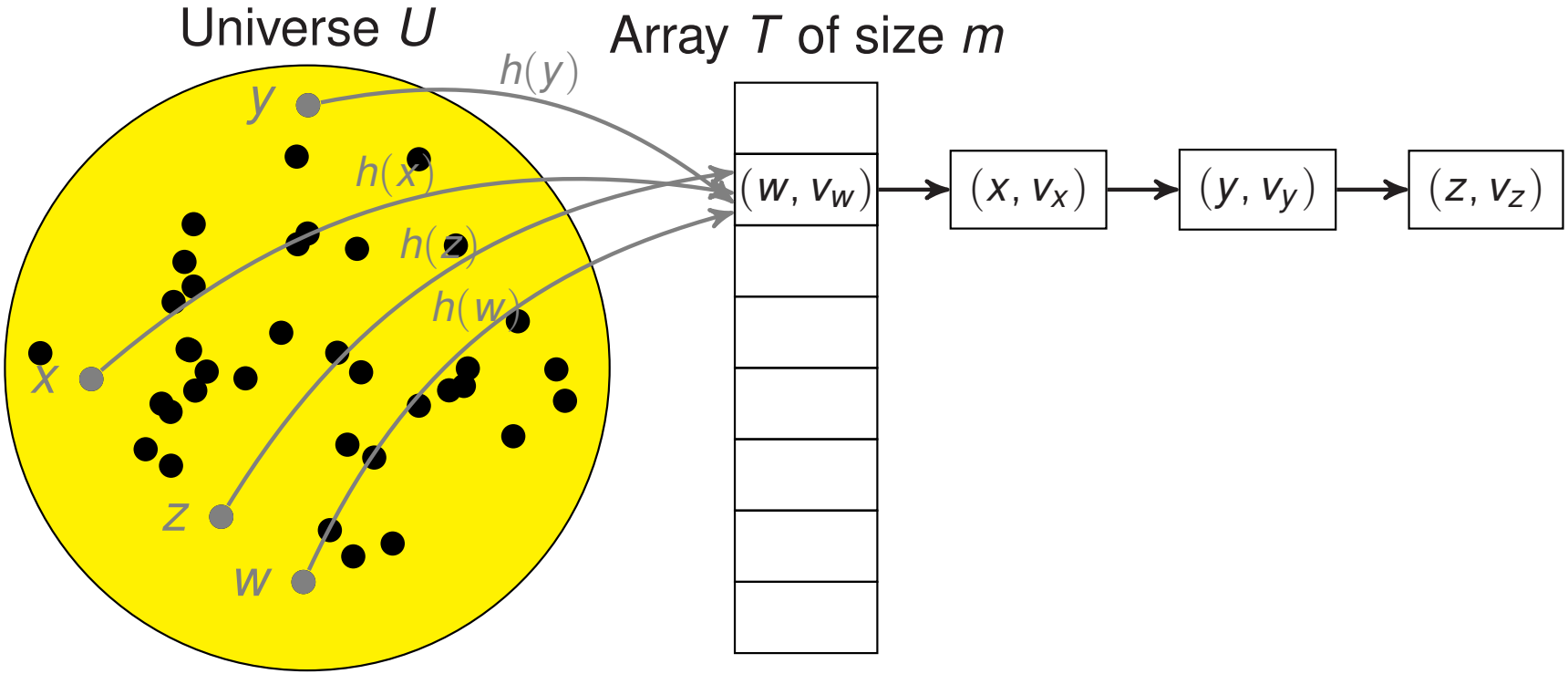
$$\Pr(Y \geq a) \leq \frac{\mathbb{E}(Y)}{a}$$

Expectation of geometric distribution

Given a sequence of Bernoulli trials with success probability p and failure probability $q = 1 - p$. Let Y be the total number of trails until the first success. Than $\mathbb{E}(Y) = \frac{1}{p}$.

Hashing revisited

- Given a set S of n keys from a universe U of size u. Usually $n \ll u$.



- A *hash function* $h : U \to [m]$ maps a key to a position in T
- T is also called *hash table*
- Goal: avoid *collisions*: $h(x) = h(y)$ for $x \neq y$
- Handle collisions by chaining

Hashing revisited

Three basic operations (and worst case complexities):

- $add(x, v_x)$: calculate $h(x)$, add item to linked list; $O(1)$ worst case
- $lookup(x)$: search linked list $L_{h(x)}$ for x; $O(|L_{h(x)}|)$ worst case
- $delete(x)$: lookup + deletion of (x, v_x) in $L_{h(x)}$; $O(1)$ on top of search time in the worst case

where (x, v_x) is a (key,value)-pair

Hashing revisited

Karlsruhe Institute of Technology

Theorem

Consider any n fixed inputs to the hash table, i.e. a sequence of add/lookup/delete operations. Pick h uniformly at random from the set of all functions $U \to [m]$. The expected run-time per operation is $O(1 + \frac{n}{m})$, or simply $O(1)$ if $n = m$.

Proof

- Let x, y be two distinct keys form U
- Let indicator r.v. $I_{x,y}$ be 1 iff $h(x) = h(y)$
- $\Pr(h(x) = h(y)) = \frac{1}{m}$ since $h(x)$ and $h(y)$ are chosen uniformly and independently from $[m]$
- Thus, $\mathbb{E}(I_{x,y}) = \frac{1}{m}$
- Let N_x be $|L_{h(x)}|$ for all keys x that are stored in T
- $N_x = \sum_{y \in T} I_{x,y}$
- $\mathbb{E}(N_x) = \mathbb{E}(\sum_{y \in T} I_{x,y}) = \sum_{y \in T} \mathbb{E}(I_{x,y}) = n \cdot \frac{1}{m} = \frac{n}{m}$

Hashing revisited

Problems:

- How to pick h uniformly at random? There are m^u hash functions from U to $[m]$. I.e. we need $u \log m$ bits to store h. That is prohibitively large.
- Choosing a fixed hash function may result in bad worst-case behavior. For $u \geq m \cdot n$ adversary can pick n keys which all map to the same position.

Solution: Universal hashing

Pick a function randomly from a set $\mathcal{H} = \{H_0, H_1, \ldots\}$ of hash functions with certain properties during the initialization of the hash table.

Hashing revisited

Universal hashing

A set $\mathcal{H}$ of hash functions is *weakly universal* if for two keys $x, y \in U$ $(x \neq y)$

$$\Pr(h(x) = h(y)) \leq \frac{1}{m}$$

where h is chosen uniformly at random from $\mathcal{H}$.

Example of a set of universal hash functions

Let m be a prime and key $x = (x_0, \ldots, x_{k-1}) \in [m]^k$. For $a = (a_0, \ldots, a_{k-1}) \in [m]^k$ define

$$h_a(x) = \sum_{i=0}^{k-1} a_i \cdot x_i \mod m$$

Then $\mathcal{H}_0 = \{h_a \mid a \in [m]^k\}$ is a universal set of hash functions.

Hashing revisited

Proof

Let $x = (x_0, \ldots, x_{k-1})$ and $y = (y_0, \ldots, y_{k-1})$ with $x \neq y$.
Count as with $h_a(x) = h_a(y)$. For each $i \neq j$ we can choose exactly one a_j with $h_a(x) = h_a(y)$:

$$\sum_{0 \leq i < k} a_i x_i \equiv \sum_{0 \leq i < k} a_i y_i \pmod{m}$$

$$\Leftrightarrow a_j \overset{(1)}{\equiv} (x_j - y_j)^{-1} \sum_{\substack{0 \leq i < k \\ i \neq j}} a_i (y_i - x_i) \pmod{m}$$

I.e. there are m^{k-1} ways to choose a such that $h_a(x) = h_a(y)$, in total there are m^k ways. $\Rightarrow \Pr(h_a(x) = h_a(y)) = \frac{1}{m}$
Question: How big is $\mathcal{H}_0$?

(1) the multiplicative inverse exists since m is prime

Static perfect hashing

Karlsruhe Institute of Technology

Problem

Given a static dictionary of n (key,value)-pairs. Devise a data structure which efficiently supports the lookup operation. Keys are from a large universe U of size u.

Solution of Fredman, Komlós and Szemerédi (J. ACM 1984)

FKS hashing scheme

- Hash table of size $O(n)$ entries
- $O(1)$ worst case lookup time
- $O(n)$ expected construction time

Static perfect hashing – first attempt

(1) Insert every key into a table of size $m = n$ using a universal hash function
(2) Check for collisions
(3) Repeat if there are collisions

How many collisions are there on average?

- For two keys x, y let $I_{x,y}$ be indicator r.v. for a collision; i.e. $I_{x,y} = 1$ iff $h(x) = h(y)$
- Let C be r.v.of total number of collisions: $\sum_{\substack{x,y \in S \\ x<y}} I_{x,y}$
-

$$\mathbb{E}(C) = \mathbb{E}\left(\sum_{\substack{x,y \in S \\ x<y}} I_{x,y}\right) = \sum_{\substack{x,y \in S \\ x<y}} \mathbb{E}(I_{x,y}) = \sum_{\substack{x,y \in S \\ x<y}} \frac{1}{m} = \binom{n}{2}\frac{1}{m} = \frac{n-1}{2}$$

- No $O(n)$ expected construction time

Static perfect hashing – second attempt

(1) Insert every key into a table of size $m = n^2$ using a universal hash function
(2) Check for collisions
(3) Repeat if there are collisions

How many collisions are there on average?

- $$\mathbb{E}(C) = \mathbb{E}(\sum_{\substack{x,y \in S \\ x<y}} I_{x,y}) = \sum_{\substack{x,y \in S \\ x<y}} \mathbb{E}(I_{x,y}) = \sum_{\substack{x,y \in S \\ x<y}} \frac{1}{m} = \binom{n}{2} \frac{1}{n^2} \leq \frac{1}{2}$$

- With Markov we get the probability of at least one collision: $\Pr(C \geq 1) \leq \frac{1}{2}$
- I.e. probability p of having no collision is $p \geq \frac{1}{2}$

Static perfect hashing – second attempt

- I.e. expected iterations for construction is 2 (with expectation of geometric distribution)
- But we do not get $O(n)$ construction time, since initialization of T takes $O(m) = O(n^2)$ time in each of the iterations.
- Question: Can you solve the issue above?
- Lookup time is $O(1)$

Static perfect hashing – third attempt

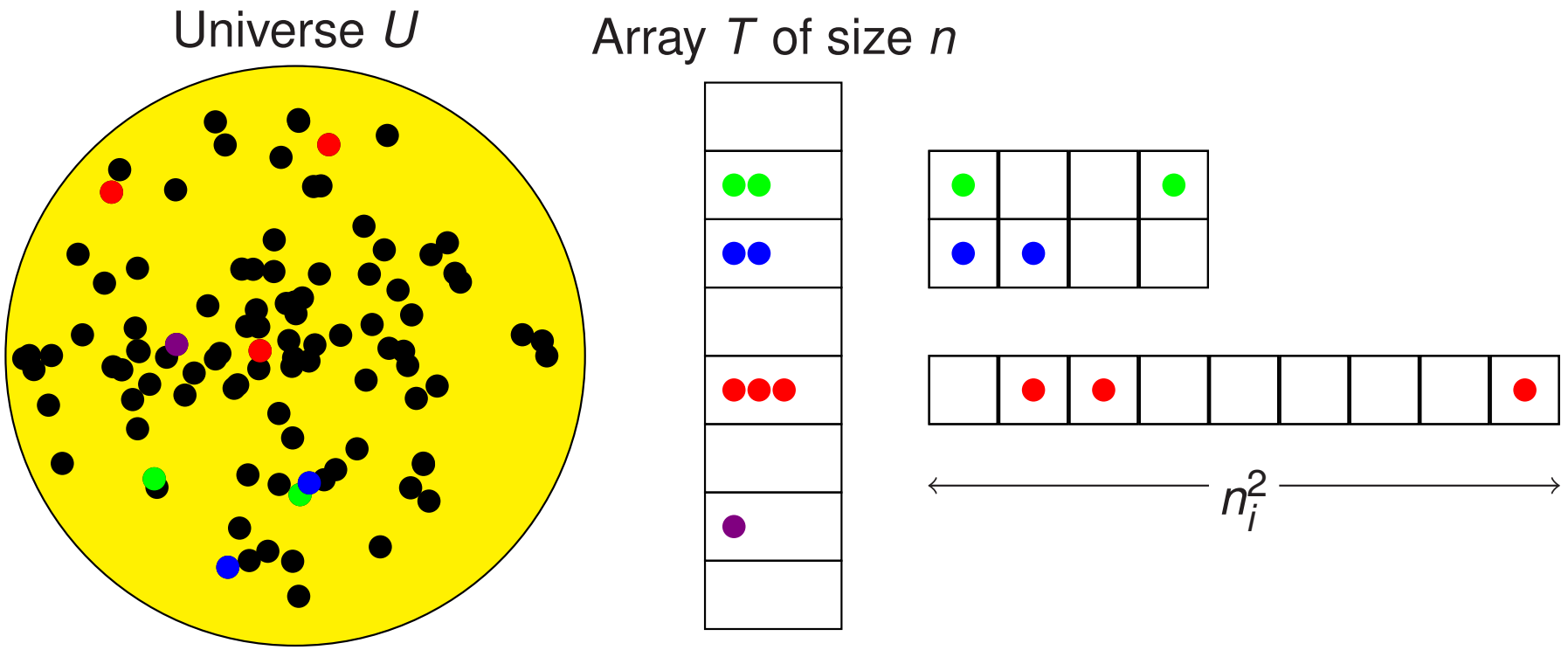
(1) Insert every key into a table of size $m = n$ using a universal hash function
(2) Check for collisions
(3) Repeat if there are more than n collisions

What is the expected construction time?

- Expected collisions: $\mathbb{E}(C) = \frac{n-1}{2} \leq \frac{n}{2}$
- The probability of at least n collisions: $\Pr(C \geq n) \leq \frac{1}{2}$
- The probability p of at most n collisions is $p \geq \frac{1}{2}$; i.e. ≤ 2 iterations are expected
- expected construction time: $O(n)$

But: lookup time could be $O(n)$

Static perfect hashing – FKS scheme



- Find hash function h to map n keys to array T of size n
- Let $n_i = |T[i]|$ be the elements in bucket i
- For each i select a hash function h_i to map the n_i keys to a table T_i of size n_i^2
- Requirement: not more than n collisions in T; no collisions in the T_is

Static perfect hashing – FKS scheme

Lookup time is constant:

00 **lookup**(x)
01 i $\leftarrow h(x)$
02 **return** $T_i[(h_i(x)]$

- What is the expected construction time?
- What is the space usage?

Static perfect hashing – FKS scheme

Space usage

- Size of T: $O(n)$ elements
- Size of T_i: $O(n_i^2)$ elements
- Size of each universal hash function: $O(\log(n))$ bits, i.e. $O(1)$ words
- Total:

$$O(n) + \sum_{i=0}^{n-1} O(n_i^2) = O(n) + O(\sum_{i=0}^{n-1} n_i^2)$$

- We know the number of collisions in T: $\sum_{i=0}^{n-1} \binom{n_i}{2} \leq n$
-

$$\sum_{i=0}^{n-1} \binom{n_i}{2} = \frac{1}{2} \sum_{i=0}^{n-1} (n_i^2 - n_i) \quad \leq \quad n$$

$$\Leftrightarrow \sum_{i=0}^{n-1} (n_i^2) \quad \leq \quad 3n$$

Static perfect hashing – FKS scheme

Space usage

- Total:

$$\begin{aligned} O(n) + \sum_{i=0}^{n-1} O(n_i^2) &= O(n) + O\Big(\sum_{i=0}^{n-1} n_i^2\Big) \\ &\leq O(n) + 3n = O(n) \end{aligned}$$

Static perfect hashing – FKS scheme

Expected construction time

- Time to construct h is $O(n)$ expected (using universal hashing)
- Time to construct h_i is $O(n_i^2)$ expected (using universal hashing)
- In total expected construction time is:

$$O(n) + \sum_{i=0}^{n-1} O(n_i^2) = O(n) + O(\sum_{i=0}^{n-1} n_i^2) = O(n)$$

Software