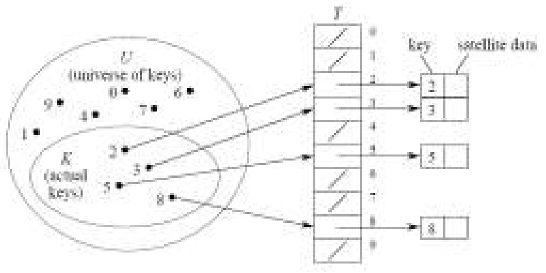
Dictionaries

- Maintain a *dynamic* set *S* of objects
  - Each object is a (*key*, *value*) pair
- Each object *x* has a unique ***key***, denoted by *x.key*, selected from universe *U,* associated with a ***value***
  - *K* is the set of keys in *S. Goal*: implement mapping *key* $\longmapsto$ *value*
- Support ***dictionary operations***:
  - Insert(*x*, *S*): stores a new (*key*, *value*) pair *x*
  - Delete(*x*, *S*), where $x \in S$: removes *x* from *S*
  - Search(*k*, *S*), where $k \in U$: returns the *value* associated with *k*
- Two efficient data structures:
  - Hashing is a generalization of simple array addressing
    - *S* stored in a table *T* of size *m*
  - Binary search trees generalize binary search
    - Allow efficient implementation of order-dependent operations

1

Direct Addressing

- One place for each item, at most one item in each place
  - Each slot corresponds to a key in *U*
  - If there is an element *x* with key *k* then *T*[*k*] points to *x*



```
DIRECT-ADDRESS-SEARCH(T, k)
return T[k]

DIRECT-ADDRESS-INSERT(T, x)
T[key[x]] ← x

DIRECT-ADDRESS-DELETE(T, x)
T[key[x]] ← NIL
```

2

Hash Tables

- Direct addressing is not practical if U is much larger than K
- Would like space to be small
  - – Use a table of size $m \in O(|K|)$
- Can still get $O(1)$ search time, but on *average*, not *worst case.*
- **Idea**: instead of storing x in slot $x.key$, use function h and store x in slot $h(x.key)$

  $h : U \to \{0,1,\ldots,m-1\}$ is the ***hash function***

  We say that k hashes to $h(k)$

3

Hash Functions

- Many different functions used in practice
  - Interpret key k as a number
- Division method
  - $h(k) = k \textbf{ mod } m$
  - If k consists of many "digits" can interpret as a $r + 1$ digit number in base b:

$$h(k) = \left(\sum_{i=0}^{r} k_i b^i\right) \textbf{mod } m$$

- Multiplication method: $h(k) = \lfloor m(k \cdot A \textbf{ mod } 1)\rfloor$, where A is a suitable constant

4

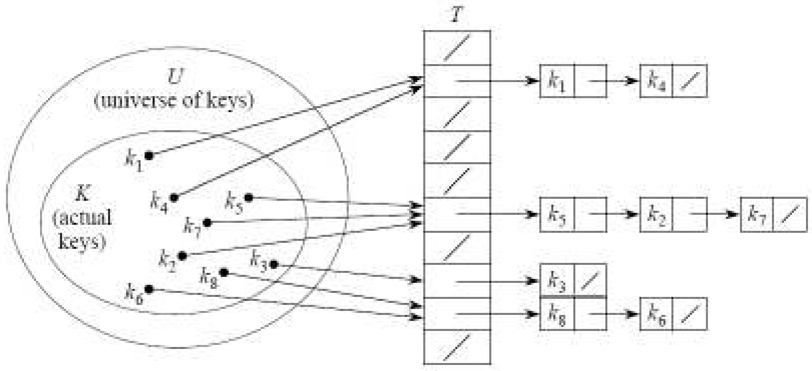
Collisions

- A ***collision*** occurs when two or more keys hash to the same slot
- Can happen whenever $|U| > 1$ or with poorly designed hashing functions
  - May or may not happen if $|K| \leq m$
  - Will definitely happen if $|K| > m$
- Can resolve by:
  - Finding a different slot (open addressing)
  - Chaining

5

Chaining

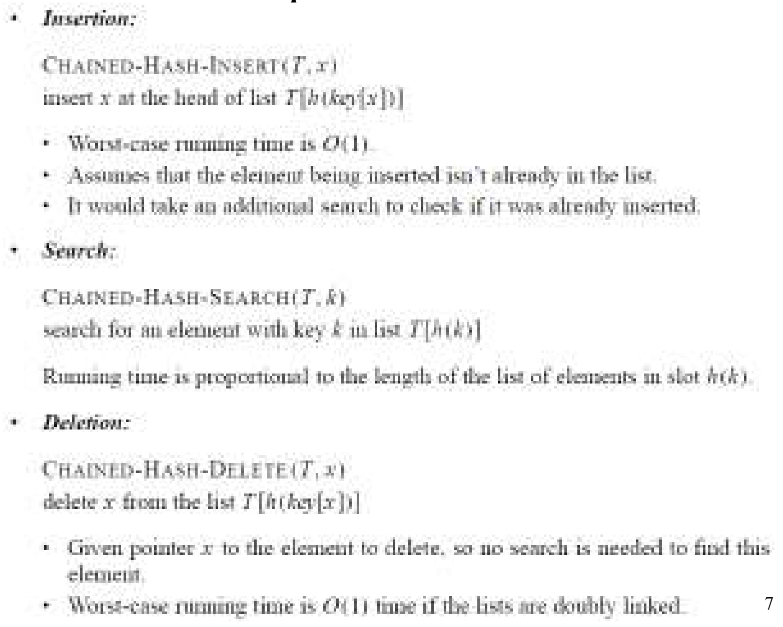
- Use a linked list to store all items that hash to the same slot.



- n_i denotes the length of the i-th chain

6

Operations



- ***Insertion:***

  CHAINED-HASH-INSERT(T, x)
  insert x at the head of list $T[h(key[x])]$

  - Worst-case running time is $O(1)$.
  - Assumes that the element being inserted isn't already in the list.
  - It would take an additional search to check if it was already inserted.

- ***Search:***

  CHAINED-HASH-SEARCH(T, k)
  search for an element with key k in list $T[h(k)]$

  Running time is proportional to the length of the list of elements in slot $h(k)$.

- ***Deletion:***

  CHAINED-HASH-DELETE(T, x)
  delete x from the list $T[h(key[x])]$

  - Given pointer x to the element to delete, so no search is needed to find this element.
  - Worst-case running time is $O(1)$ time if the lists are doubly linked.

7

Performance

- Analysis assumes ***simple uniform hashing***:

  any element is equally likely to hash to any of the m slots

- In terms of ***load factor*** $\alpha = n/m$

Theorem. With simple uniform hashing, a search (successful or not) runs in expected $\Theta(1+\alpha)$ time.

Why is this result not useful enough?

8

Universal Hashing: Motivation

- Any fixed *h* is vulnerable to "malicious adversary", i.e., there is a set of keys that makes that *h* perform poorly.
- Can improve performance by randomly choosing h, independently of *K*.
- No set *K* can consistently elicit worst case behavior.
- Algorithm may perform differently on different runs with same input.

9

Universal Hashing

Definition. A collection of hash functions $\mathcal{H} = \{h \text{ where } h: \mathcal{U} \rightarrow \{0, \dots, m-1\}\}$ is ***universal*** if for every pair of keys $i, j \in \mathcal{U}$, $i \neq j$, the number of functions $h \in \mathcal{H}$ for which $h(i) = h(j)$ is $\leq |\mathcal{H}|/m$

Theorem. Let *h* be chosen at random from a universal set and used to hash *n* keys, with chaining, into a table *T* of size m. For any $k \in \mathcal{U}$, the expected length of chain $T[h(k)]$ is at most α if $k \notin K$ and at most $1 + \alpha$ if $k \in K$.

10

Proof. For every $i, j, k \in U$, define:

$$X_{ij} = I[h(i) = h(j)] \text{ and } Y_k = \sum_{j \in K, j \neq k} X_{kj}$$

Then, $$E[Y_k] = E\left[\sum_{h \in K, h \neq k} X_{kh}\right] = \sum_{h \in K, h \neq k} E[X_{kh}] \leq \sum_{h \in K, h \neq k} \frac{1}{m}$$

Two cases:

1. $k \notin K \Rightarrow n_{h(k)} = Y_k \Rightarrow E[n_{h(k)}] \leq n/m = \alpha$
2. $k \in K \Rightarrow n_{h(k)} = 1 + Y_k \Rightarrow E[n_{h(k)}] \leq 1 + (n-1)/m < 1 + \alpha$

11

Universal Hashing…

Theorem. Consider a chained hash table with m slots built using universal hashing. If the number of insertions is $O(m)$ then the expected time to process any sequence of n Insert, Search, and Delete operations is $\Theta(n)$.

12

Fields

- A field is a set F with two binary operations $\oplus$ ("addition") and $\odot$ ("multiplication") that satisfies the following properties:

  Closure: $\forall\ a, b \in F:\ a \oplus b \in F,\ a \odot b \in F$

  Commutativity: $\forall a, b \in F,\ a \oplus b = b \oplus a,\ a \odot b = b \odot a$

  Associativity of addition: $\forall a, b, c \in F,\ (a \oplus b) \oplus c = a \oplus (b \oplus c)$

  Additive Identity: $\exists\ z \in F : \forall\ a \in F : a \oplus z = z \oplus a = a$

  Additive Inverse: $\forall\ a \in G : \exists\ b \in G : a \oplus b = b \oplus a = z$

  Associativity of multiplication: $\forall a, b, c \in F, (a \odot b) \odot c = a \odot (b \odot c)$

  Multiplicative Identity: $\exists\ u \in F : \forall\ a \in F : a \odot u = u \odot a = a$

  Multiplicative Inverse: $\forall a \in F, a \neq z : \exists b \in F:\ a \odot b = b \odot a = u$

  Distributivity: $\forall a, b, c \in F,\ a \odot (b \oplus c) = (a \odot b) \oplus (a \odot c),$
  $(b \oplus c) \odot a = (b \odot a) \oplus (c \odot a)$

- A set with one operation that satisfies closure, associativity, and existence of an identity and inverses is called a *group*.
  - The set of nonzero elements of a field from a group with respect to $\odot$

The Field $\mathbb{Z}_p$

- Consider the integers $\{0, 1, \ldots, p-1\}$ with the usual addition and multiplication modulo p
- If p is prime, then $\left(Z_p, +\text{mod } p, *\text{ mod } p\right)$ is a field

$\mathbb{Z}_5^+$	**0**	**1**	**2**	**3**	**4**
0	0	1	2	3	4
1	1	2	4	4	0
2	2	3	4	0	1
3	3	4	0	1	2
4	4	0	1	2	3

$\mathbb{Z}_5^*$	**1**	**2**	**3**	**4**
1	1	2	3	4
2	2	4	1	3
3	3	1	4	2
4	4	3	2	1

We denote the multiplicative inverse of a by a^{-1}, e.g., in $\mathbb{Z}_5$ $3^{-1} = 2$

$\mathbb{Z}_q^*$, for non prime q

- If q is not prime, then $\mathbb{Z}_q^*$ is not a group

$\mathbb{Z}_6^*$	1	2	3	4	5
1	1	2	3	4	5
2	2	4	**0**	2	4
3	3	**0**	3	**0**	3
4	4	2	**0**	4	2
5	5	4	3	2	1

$\mathbb{Z}_7^*$	1	2	3	4	5	6
1	1	2	3	4	5	6
2	2	4	6	1	3	5
3	3	6	2	5	1	4
4	4	1	5	2	6	3
5	5	3	1	6	4	2
6	6	5	4	3	2	1

15

Constructing a Universal Set

- Consider a universe of keys $\mathcal{U} = \{0, \ldots, u-1\}$ and let p be a prime such that $u \leq p \leq 2u$
- Assume $m < p$ (else use direct addressing!)
- For $a \in \mathbb{Z}_p^*$ and $b \in \mathbb{Z}_p$ define a hashing function $h_{ab}(k) = ((a \cdot k + b) \bmod p) \bmod m$
- The family of hashing functions is defined as $\mathcal{H}_{pm} = \{h_{ab} : a \in \mathbb{Z}_p^* \text{ and } b \in \mathbb{Z}_p\}$
- Size m of table is arbitrary provided $m < p$
- How big is $\mathcal{H}_{pm}$? $p(p-1)$

16

Universality

Theorem. The class $\mathcal{H}_{pm}$ is universal

Proof. Let $k, \ell \in \mathcal{U}$, $k \neq \ell$ and $a \in \mathbb{Z}_p^*, b \in \mathbb{Z}_p$

Let $r = ak + b \bmod p$ and $s = a\ell + b \bmod p$

Then $r \neq s$ (no collision at the mod p level). Why?

Each of the $p(p-1)$ choices of h_{ab} yields a different pair $(r, s) \Rightarrow$ we have a bijection between pairs (a, b) and pairs (r, s). Why?

$$a = (r - s)(k - \ell)^{-1} \bmod p \text{ and } b = (r - ak) \bmod p$$

$\Pr\left(h_{ab}(k) = h_{ab}(\ell)\right) = \Pr(r = s \bmod m)$ where *r* and *s* are random distinct values in $\mathbb{Z}_p$ (each pair equally likely)

For fixed *r*, how many $s \neq r$ result in $s = r \bmod m$?

17

Universality Proof…

For a given *r*, how many $s \neq r$ result in $s = r \bmod m$?

Answer: at most $\lceil p/m \rceil$ values from $\mathbb{Z}_p$ are $= 0 \bmod p$. Since $s \neq r$, at most $\lceil p/m \rceil - 1$ of the remaining values result in $s = r \bmod p$

However, $\lceil p/m \rceil - 1 \leq \left(\frac{p+m-1}{m}\right) - 1 = \frac{p-1}{m}$

Pr(collision) $= \Pr[h_{ab}(k) = h_{ab}(\ell)]$

$$\leq \frac{(p-1)/m}{p-1} = \frac{1}{m}$$

18

A Different Universal Set

- Table size m is a prime number.
- Each key x consists of r+1 digits, base m

$$x = \langle x_r \cdots x_1 x_0 \rangle$$

- Choose random (r+1)-digit base-m number a:

$$a = \langle a_r \cdots a_1 a_0 \rangle$$

$$h_a(x) = \sum_{i=0}^{r} a_i x_i \bmod m$$

Theorem. $H = \{h_a\}$ is universal.

Proof. For x,y with $x_0 \neq y_0$, <u>arbitrary</u> $a_1, a_2, \ldots, a_r$, how many a_0 satisfy

$$a_0(x_0 - y_0) = \sum_{i=1}^{r} a_i(y_i - x_i) \bmod m$$

19

Number of Solutions to $a_0\, b = w$

- Can assume $b \neq 0$
- Two cases
  1. $w = 0 \Rightarrow a_0 = 0$ is the only solution
  2. $w \neq 0 \Rightarrow a_0 = w\, b^{-1}$ is the only solution
- Either way, the number of hashing functions that result in a collision of x and y is m^r
- Therefore, for $i \neq j \in \mathcal{U}$, when h is chosen randomly from $\mathcal{H}$, $\Pr[h(i) = h(j)] = 1/m$

20

Perfect Hashing

Goal: Given a set K of n keys, construct a *static* hash table of size $m = O(n)$ such that Search takes $O(1)$ time.

- *Every* search takes $O(1)$ time in the worst case
- Statistical variation from list to list or key to key does not affect performance

Definition. A perfect hash function maps different elements of K to different slots, i.e., it is a *total injective function*

Applications: compiler keywords, 10000 most common words in the English dictionary, files on a CD.

21

Markov's Inequality

Theorem. If X is a non-negative random variable then $P[X \geq t] \leq E[X] / t$

Proof. Define an indicator variable $Y = \mathrm{I}(X \geq t)$. Then, $P(X \geq t) = E(Y)$. Since $Y \leq X/t$ for all t, then $P(X \geq t) = E(Y) \leq E(X/t) = E(X) / t$

22

Collision Free Hashing

Theorem. Suppose you store n keys in a hash table of size $m \geq n^2$ using a hash function randomly chosen from a universal set. Then, the probability of having any collisions is less than 1/2

Proof. Let X denote the number of colliding pairs.
There are $n(n-1)/2$ pairs that may collide.
Each pair collides with probability $\leq 1/m$
$E[X] \leq n(n-1)/2m \leq n(n-1)/2n^2 < 1/2$
By Markov's inequality (with t=1) $P[X \geq 1] \leq E[X] < 1/2$

23

A Better Idea

- Quadratic storage for a large set of keys is not reasonable
- Instead, use *2-level hashing* with universal hashing at both levels.
  - No chaining, instead have m secondary hash tables built with universal hashing
  - Each secondary table has size $m_i = (n_i)^2$ where n_i is the number of keys in slot i
- May have collisions at level 1 but no collisions at level 2

24

Theorem. Suppose we store n keys into a hash table of size $m \geq n$, using a hash function h chosen at random from a universal set of hash functions. Then

$$E\left[\sum_{j=0}^{m-1} n_j^2\right] < 2n$$

Proof.

$$E\left[\sum_{j=0}^{m-1} n_j^2\right] = E\left[\sum_{j=0}^{m-1}\left(n_j + 2\binom{n_j}{2}\right)\right] \quad \left(\text{because } a^2 = a + 2\binom{a}{2}\right)$$

$$= E\left[\sum_{j=0}^{m-1} n_j\right] + 2E\left[\sum_{j=0}^{m-1}\binom{n_j}{2}\right] \quad (\text{linearity of expectation})$$

$$\leq n + 2\binom{n}{2}\frac{1}{m} \quad (\text{expected number of collisions})$$

$$\leq n + 2\frac{n-1}{2} = 2n - 1 \quad (\text{because } m \geq n)$$

Corollary. Suppose we store n keys into a hash table of size $m \geq n$, using a hash function h chosen at random from a universal set and we set the size of each secondary table to $m_j = (n_j)^2$, $j = 0,\ldots m-1$. Then

1. The expected total storage required by the secondary hash tables is $< 2n$
2. The probability is $< 1/2$ that the total storage required by the secondary tables is $\geq 4n$

Proof.

$$1.\ E\left[\sum_{j=0}^{m-1} m_j\right] = E\left[\sum_{j=0}^{m-1} n_j^2\right] < 2n$$

$$2.\ \Pr\left[\sum_{j=0}^{m-1} m_j \geq 4n\right] \leq \frac{E\left[\sum_{j=0}^{m-1} m_j\right]}{4n} < \frac{2n}{4n} = \frac{1}{2}$$