HASH TABLES

CS340

Division Method Hash Function

- H(k) = k mod m (k is number of keys, m is number of slots)

- **Example**: m = 20 and k = 91. h(k) = 11.
- **Advantage**: Fast, since requires just one division operation.
- **Disadvantage**: Have to avoid certain values of m:
  - Powers of 2 are bad. If $m = 2^p$ for integer p, then h(k) is just the least significant p bits of k.
- **Good choice for m**: A prime not too close to an exact power of 2.

Multiplication method hash function

- 1. Choose constant A in the range $0 < A < 1$.
- 2. Multiply key k by A.
- 3. Extract the fractional part of kA.
- 4. Multiply the fractional part by m.
- 5. Take the floor of the result.
- **Disadvantage:** Slower than division method.
- **Advantage:** Value of m is not critical.

Open Addressing

- Store all keys in the hash table itself.
- Each slot contains either a key or NIL.
- To search for key k:
  - Compute h(k) and examine slot h(k). Examining a slot is known as a probe.
  - If slot h(k) contains key k, the search is successful. If this slot contains NIL, the search is unsuccessful.
  - There's a third possibility: slot h(k) contains a key that is not k. We compute the index of some other slot, based on k and on which probe (count from 0: 0th, 1st, 2nd, etc.) we're on.
  - Keep probing until we either find key k (successful search) or we find a slot holding NIL (unsuccessful search).

Sequence of slots probed

- We need the sequence of slots probed to be a permutation of the slot numbers (so that we examine all slots if we have to, and so that we don't examine any slot more than once).
- To insert, act as though we're searching, and insert at the first NIL slot we find.

Open Addressing

```
HASH-SEARCH(T, k)
  i = 0
  repeat
      j = h(k, i)
      if T[j] == k
          return j
      i = i + 1
  until T[j] == NIL or i = m
  return NIL
```

```
HASH-INSERT(T, k)
  i = 0
  repeat
      j = h(k, i)
      if T[j] == NIL
          T[j] = k
          return j
      else i = i + 1
  until i == m
  error "hash table overflow"
```

Probing Techniques: Linear

- Given auxiliary hash function h', the probe sequence starts at slot h'(k) and continues sequentially through the table, wrapping after slot m - 1 to slot 0.
- Given key k and probe number i (0 <= i < m), h(k,i)= (h'(k) + i) mod m.
- **DISADVANTAGE:** Linear probing suffers from primary clustering: long runs of occupied sequences build up.

Interview Questions

- Consider a hash table of size 7 with hash function:
  h(k)= k mod 7
- Insert in order 19, 26,13, 48,17
- What is result when collisions are handled by linear probing

Probing Techniques: Quadratic

- As in linear probing, the probe sequence starts at h'(k).
- Unlike linear probing, it jumps around in the table according to a quadratic function of the probe number:
  - $h(k,i) = (h'(k) + c_1 i + c_2 i^2) \bmod m$, where c1, c2 != 0 are constants.
- For example, if $h(k,i) = (h'(k) + i + i^2) \bmod m$, the sequence will be: h(k), h(k)+2, h(k)+6…

Probing Techniques: Double hashing

- Use two auxiliary hash functions, h1 and h2. h1 gives the initial probe, and h2 gives the remaining probes: $h(k, i) = (h1(k) + ih2(k)) \bmod m$.

Interview Questions

- Consider a hash table of size 7 with hash function:
  $h(k) = k \bmod 7$
- Insert in order 19, 26,13, 48,17
- When collisions are handled by double hashing where the second hash function is $h'(k) = 5 - (k \bmod 5)$.

- Insert in order 19, 26,13, 48,17
- H(k,i) = (k mod 7 + i * [5 – k mod 5]) mod 7 =