CMSC 341
Lecture 12

Announcements
Postponing Priority Queues until later
HashTable Class

```cpp
template <class HashedObj>
class HashTable {
public:
    explicit HashTable(const HashedObj &notFound, size=101);
    HashTable(const HashTable &rhs) :
        ITEM_NOT_FOUND(rhs.ITEM_NOT_FOUND),theLists(rhs.theLists){
    }
    const HashedObj &find(const HashedObj &x) const;
    void makeEmpty();
    void insert (const HashedObj &x);
    void remove (const HashedObj &x);
    const HashTable &operator=(const HashTable &rhs);
private:
    vector<List<HashedObj>> theLists;
    const HashedObj ITEM_NOT_FOUND;
};
```
Handling Collisions Revisited

Open addressing (aka closed hashing)
- all elements stored in the table itself (so table should be large. Rule of thumb: M>= 2N)
- upon collision, item is hashed to a new (open) slot.

Hash function
\[ h: U \times \{0,1,2,\ldots\} \rightarrow \{0,1,\ldots,M-1\} \]
\[ h(k,I) = (h'(k) + f(I)) \mod m \]
for some \( h': U \rightarrow \{0,1,\ldots,M-1\} \)
and \( f(0) = 0 \)

Each try is called a **probe**

Linear Probing

Function:
\[ f(i) = ci \]

Example:
\[ h'(k) = k \mod 10 \] in a table of size 10 , \( f(i) = i \)
\[ U=\{89,18,49,58,69\} \]
Linear Probing (cont)

Problem: Clustering
– when table starts to fill up, performance $\rightarrow O(N)$

Asymptotic Performance
– insertion and unsuccessful find, average
  • # probes $\equiv \frac{1}{2}(1+1/(1-\lambda)^2)$
  • if $\lambda \equiv 1$, the denominator goes to zero and the number of probes goes to infinity

Linear Probing (cont)

Remove
– Can’t just use the hash function(s) to find the object, and remove it, because objects that were inserted after $x$ were hashed based on $x$’s presence.
– Can just mark the cell as deleted so it won’t be found anymore.
  • Other elements still in right cells
  • Table can fill with lots of deleted junk
Quadratic Probing

Function:
\[ f(i) = c_2 i^2 + c_1 i + c_0 \]

Example:
\[ f(i) = i^2, \ m=10 \]
\[ U=\{89,18,49,58,69\} \]

Quadratic Probing (cont.)

Advantage:
- reduced clustering problem

Disadvantages:
- reduced number of sequences
- no guarantee that empty slot will be found if table size is not prime
Double Hashing

Use two hash functions: \( h'_1(k), h'_2(k) \)
\[
h(k, l) = (h'_1(k) + lh'_2(k)) \mod M
\]
Choosing \( h'_2(k) \)
- don’t allow \( h'_2(k) = 0 \) for any \( k \).
- a good choice:
  - \( h'_2(k) = R - (k \mod R) \) with \( R \) a prime smaller than \( M \)

Characteristics
- No clustering problem
- Requires a second hash function

Balanced Trees

Problem
- specific tree configuration dependent on order in which nodes are inserted
- may become noticeably unbalanced, leading to performance approaching worst case -- O(n)

Solution
- ensure that trees remain balanced (or pretty balanced), no matter what
AVL Trees

Method
– impose balance condition: The height of the left and right subtrees of a node can differ by no more than one.
– adjust structure after each insertion or deletion to maintain balance condition; uses rotation

Pros and Cons
– guarantee O(lg n) performance
– lots of adjustments can make insertion expensive

Rotation Operation

To rotate about a node t and its left child l:
\[ t->left = l->right; \]
\[ l->right = t; \]

To rotate about a node t and its right child r:
\[ t->right = r->left; \]
\[ r->left = t; \]
Splay Trees

Concept
– adjust tree in response to accesses to make common operations efficient
– after access node is moved to root by splaying

Performance
– amortized such that m operations take $O(m \log n)$ where n is the number of insertions

Splay Operation

Traverse tree from node x to root, rotating along the way until x is the root

Each rotation
– If x is root, do nothing.
– If x has no grandparent, rotate x about its parent.
– If x has a grandparent,
  • if x and its parent are both left children or both right children, rotate the parent about the grandparent, then rotate x about its parent
  • if x and its parent are opposite type children (one left and the other right), rotate x about its parent, then rotate x about its new parent (former grandparent)
This EPS picture was not saved with a preview included in it. 
Comment: 
This EPS picture will print to a PostScript printer, but not to 
other types of printers.