CSE 247
Data Structures

Abstract Data Type: Priority Queue

- A priority queue is a collection of zero or more items and each item is associated with a priority

- A priority queue has at least three operations
  - insert(item i) (enqueue) a new item
  - delete() (dequeue) the member with the highest priority
  - find() the item with the highest priority

- Note that in a priority queue "first in first out" does not apply in general.

Priority Queues

- Often the items added to a queue have a *priority* associated with them: this priority determines the order in which they exit in the queue - highest priority items are removed first.
- This situation arises often in process control systems. Imagine the operator's console in a large automated factory.
- occasionally something breaks or fails, and alarm messages are sent. These have high priority because some action is required to fix the problem

Application

- Simulation of automobile traffic on street.
- Patient queue, emergency cases with high priority
- Event scheduling
- Simulation of Parking lot and garage
- Path finding algorithm (Dijkstra's algorithm, A* search algorithm)
- Data compression (Huffman coding)
- Heap sort

Priority Queue

- The priority queue is a data structure in which the natural ordering of the elements does determine the results of its basic operations.
- Two types of priority queue
  - Ascending
  - Descending
- Stack can be view as descending priority queue, whose element are ordered by time of insertion.
- Queue is ascending priority queue.

Array implementation of priority Queue

- The delete operation in ascending priority queue. This raise two issues
    - Locate smallest element, Every element of the array must be examined.
    - How element in the middle of array be deleted.
- Priority queue deletion requires both searching and movement.

Solutions

1. An empty indicator can be placed into deleted position. The indicator can be a value or separate field.

   Disadvantages:

   - search process to locate max or min examine all deleted positions.

2. Each deletion compact the array by shifting all elements.

   Disadvantage:

   - the deletion become so inefficient.

3. Maintain an array as an ordered array. this method moves the work of searching and shifting from deletion operation to insertion.

Solution

- In the last solution array is in sorted order according to priority so no search operation is required at the time of deletion.

Disadvantage of linked list

- Link list occupies more storage than a corresponding element in an array.
- Time spent in Management of available list

Priority Queue

1. Handle priority at insertion then:
   - Insert element in a queue according to priority takes O(n).
   - Deletion takes O(1).

2. Handle priority at deletion then:
   - Insert takes O(1).
   - Deletion require searching highest priority element that takes O(n).