Linked Lists, Stacks, and Queues

Dr. Baldassano

chrisb@princeton.edu

Yu’s Elite Education
Last week recap

- Heaps
  - Heap structure and ordering
  - Adding elements
  - Removing elements
  - Using as priority queue
  - Heapsort
Problems with arrays

- Traditional arrays/lists store data in consecutive slots in memory
- Good: can easily calculate where element \(i\) can be found, as \((\text{begin} + i \times \text{datasize})\)
- Bad: Inserting or deleting elements requires moving \(O(N)\) elements (unless at end)
- Python example
Linked lists

- Linked lists do not store elements consecutively in memory
- Each element contains some data, and the memory address of the next element

http://visualgo.net/list.html
Big-O for linked lists

- Inserting or deleting a node (assuming we know memory address of insertion/deletion point) is $O(1)$ (vs. $O(n)$ for array)
- Indexing (finding nth piece of data) is $O(n)$ (vs. $O(1)$ for array)
Implementing linked list

- Node structure
  - Data field
  - Pointer to next node (NULL if end of list)
- Head pointer to first node

- Example in python
Reversing linked list

- Can we reverse all links in a linked list in a single pass?
Detecting loops in a linked list

- How can we check that our list doesn’t have any loops?
- Traverse until end?
- Mark each node?
- Check node against entire list so far?
  - Works, but $\mathcal{O}(N^2)$
- Floyd’s tortoise and hare
  - [http://visualgo.net/cyclefinding.html](http://visualgo.net/cyclefinding.html)
Problems with one-directional links

- We can only iterate through the list in one direction
- Even if given the memory address of a node to delete, we’d have to iterate through the list to find the previous node

- Solution: Doubly-linked lists
Doubly-linked lists

- Every node contains data, and pointers to both previous and next nodes
- Have head and tail pointers to the front and back of the list
- Example in python (for show and insert)
Circular doubly-linked lists

- We can get rid of some of the special cases at the ends of the list by connecting the head and tail nodes to each other

Example in python (for show and insert)
Stacks and Queues

- Linked lists are often used to implement two other data structures: stacks and queues.
- Both allow us to add or remove one element at a time.
- Queues: First-in-first-out (FIFO)
- Stacks: First-in-last-out (FILO)
Uses of queues?

- Website requests
- Simulations or game engines
- Expanding algorithms like Dijkstra's
Stacks

Uses of stacks?

- Recursive algorithms like mergesort
- Backtracking algorithms like constraint satisfaction (e.g. Sudoku)
- Compiling and running programs
Implementing Queue with linked list

- Why implement with linked list?
- Enqueue operation:
  - Add node at head
- Dequeue operation
  - Remove and return node at tail
- [http://visualgo.net/list.html](http://visualgo.net/list.html)
Implementing Stack with linked list

- Why implement with linked list?
- Push operation:
  - Add node at head
- Pop operation:
  - Remove and return node at head
- [http://visualgo.net/list.html](http://visualgo.net/list.html)
Stack example:
Reverse Polish Notation

- Traditional math notation: requires knowing order of operations, using parentheses
  \[7 + (1 + 2) \times 5 = 22\]
- Reverse Polish notation avoids these issues by using a stack
  \[7 1 2 + 5 * +\]
- Implement in python
Assignment: Median of sorted circularly linked list

- Sorted singly-linked circular list:

- Given a reference to some node, find median of the list
  - Note: need to handle all-equal special case