403: Algorithms and Data Structures

Heapsort and Priority Queues

Fall 2016

UAlbany

Computer Science

Some slides borrowed from David Luebke

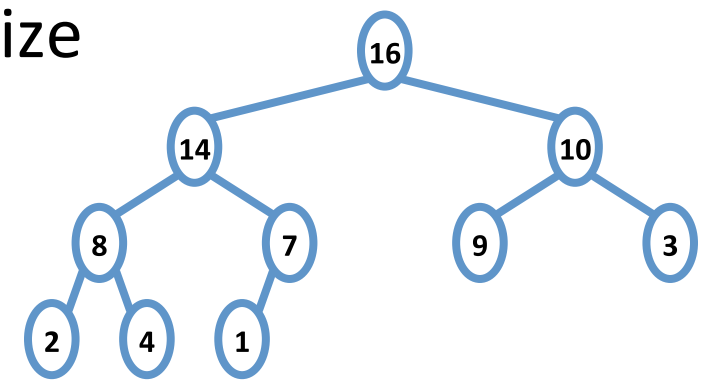
Context

- We defined heaps
  - ”almost” complete binary trees
  - **$A[\mathit{Parent}(i)] \geq A[i]$** for all nodes **$i > 1$**
- Heap operations: Heapify()
  - Fix a single violation of the heap property
  - “Float” values down the tree
  - O(logn), where n is the heap size
    - What is the base of the log?

A =

16	14	10	8	7	9	3	2	4	1

=



Heap Operations: BuildHeap()

- Input: Array **A[1…n]**
- Required: Convert **A** into a heap
- Idea: build a heap in a bottom-up manner by running **`Heapify()`** on successive subarrays

Heap Operations: BuildHeap()

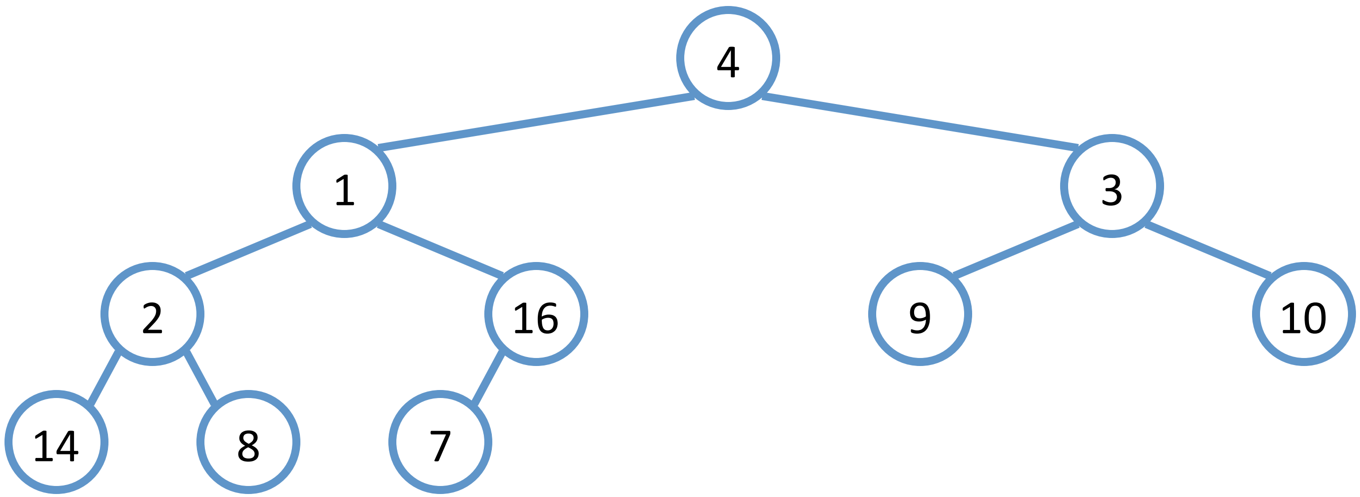
- Fact: for array of length *n*, all elements in range A[$\lfloor n/2 \rfloor$ + 1 .. n] are leaves (*Why?*)
  - Left($\lfloor n/2 \rfloor$ + 1) = 2* ($\lfloor n/2 \rfloor$ + 1) > n
- Another fact: Leaves are (trivially) heaps
- Walk backwards through the array from $\lfloor n/2 \rfloor$ to 1, calling **`Heapify()`** on each node.
  - Order of processing guarantees that the children of node *i* are heaps when *i* is processed
  - Why is this important?

BuildHeap()

```
// given an unsorted array A, make A a heap
BuildHeap(A)
{
  heap_size(A) = length(A);
  for (i = ⌊length[A]/2⌋ downto 1)
     Heapify(A, i);
}
```

BuildHeap() Example

- Work through example
  A = {4, 1, 3, 2, 16, 9, 10, 14, 8, 7}



Crude Analysis of BuildHeap()

- Each call to **`Heapify()`** takes O(lg n) time
- There are O(n) such calls (specifically, $\lfloor n/2 \rfloor$)
- Thus the running time is O(n lg n)
  - *Is this a correct asymptotic upper bound?*
  - *Is this an asymptotically tight bound?*
- A tighter bound is O(n)
  - *How can this be? Is there a flaw in the above reasoning?*

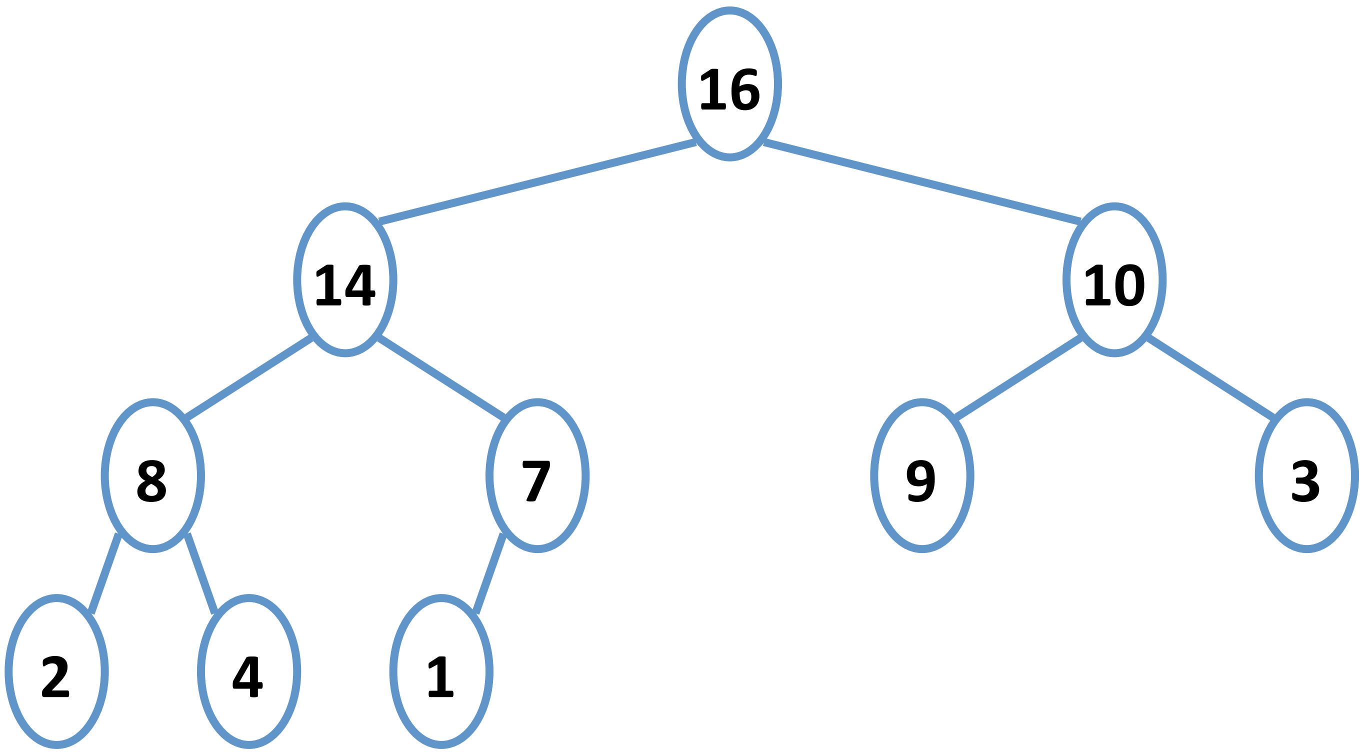
Analyzing BuildHeap(): Tight

- To **`Heapify()`** a subtree takes O(h) time where h is the height of the subtree
  - h = O(lg m), m = # nodes in subtree
  - Intuition: The height of most subtrees is small , i.e. O(log n) is too "generous"
- Fact 1: an n-element heap has at most $\lceil n/2^{h+1} \rceil$ nodes of height h
  - *Proof?*
- Using Fact 1 we can show that **`BuildHeap()`** takes O(n) time
  - Proof?

Heapsort

- Given **`BuildHeap()`**, an in-place sorting algorithm is easily constructed:
  - Maximum element is at A[1]
  - Discard by swapping with element at A[n]
    - Decrement heap_size[A]
    - A[n] now contains correct value
  - Restore heap property at A[1] by calling **`Heapify()`**
  - Repeat, always swapping A[1] for A[heap_size(A)]

Example



Heapsort

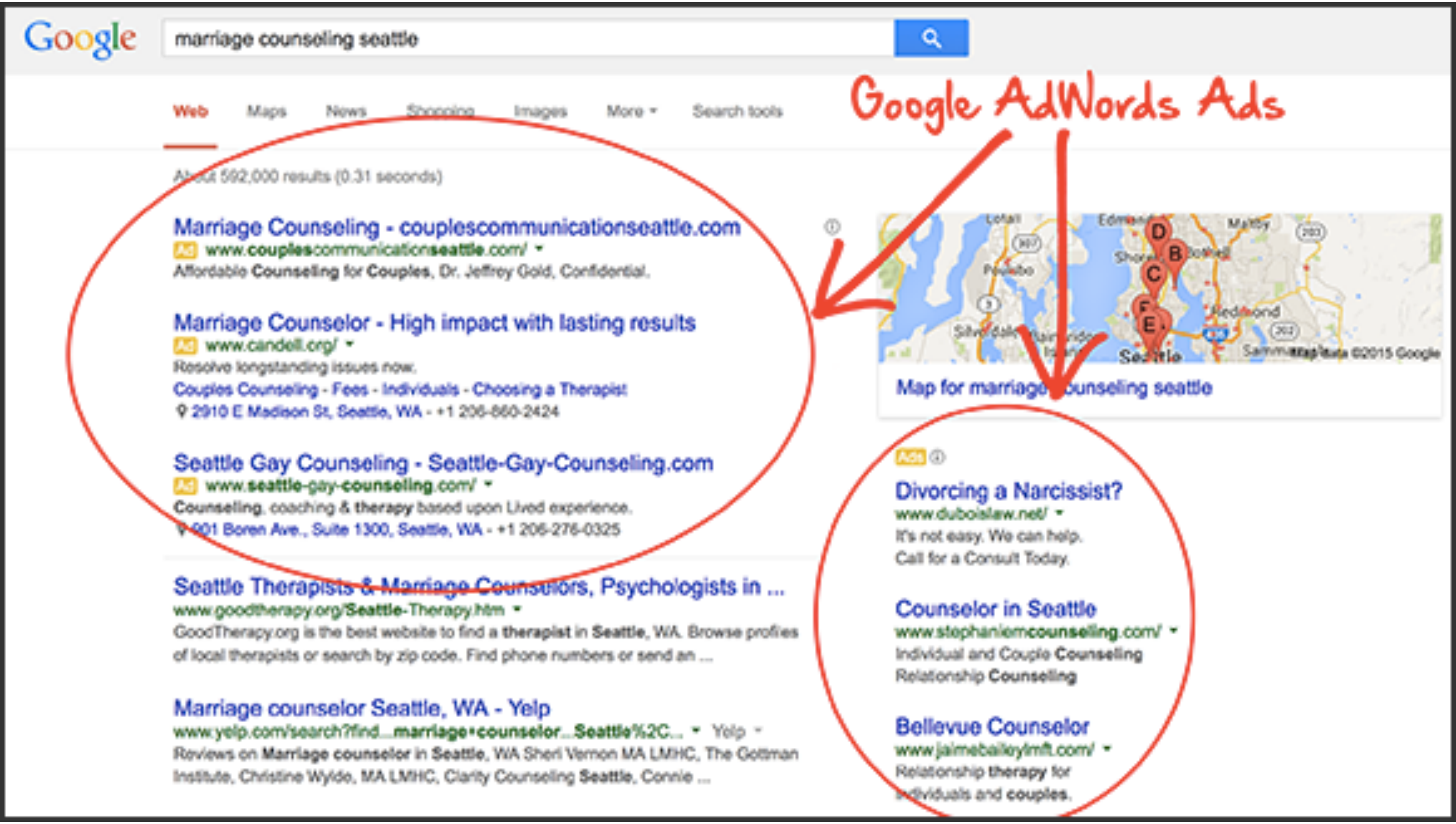
```
Heapsort(A)
{
   BuildHeap(A);
   for (i = length(A) downto 2)
   {
      Swap(A[1], A[i]);
      heap_size(A) -= 1;
      Heapify(A, 1);
   }
}
```

Analyzing Heapsort

- The call to **`BuildHeap()`** takes O(n) time
- Each of the n - 1 calls to **`Heapify()`** takes O(lg n) time
- Thus the total time taken by **`HeapSort()`**
  = O(n) + (n - 1) O(lg n)
  = O(n) + O(n lg n)
  = O(n lg n)

Priority Queues

- Heapsort is a nice algorithm, but in practice Quicksort (coming up) usually wins
- But the heap data structure is incredibly useful for implementing *priority queues*
  - A data structure for maintaining a set *S* of elements, each with an associated value or *key*
  - Supports the operations **`Insert()`**, **`Maximum()`**, and **`ExtractMax()`**
  - ***What might a priority queue be useful for?***


Google
marriage counseling seattle
Web
Maps
News
Shopping
Images
More
Search tools
Google AdWords Ads
About 592,000 results (0.31 seconds)
Marriage Counseling - couplescommunicationseattle.com
Ad www.couplescommunicationseattle.com/
Affordable Counseling for Couples, Dr. Jeffrey Gold, Confidential.
Marriage Counselor - High impact with lasting results
Ad www.candell.org/
Resolve longstanding issues now.
Couples Counseling - Fees - Individuals - Choosing a Therapist
2910 E Madison St, Seattle, WA - +1 206-860-2424
Seattle Gay Counseling - Seattle-Gay-Counseling.com
Ad www.seattle-gay-counseling.com/
Counseling, coaching & therapy based upon Lived experience.
901 Boren Ave., Suite 1300, Seattle, WA - +1 206-276-0325
Seattle Therapists & Marriage Counselors, Psychologists in ...
www.goodtherapy.org/Seattle-Therapy.htm
GoodTherapy.org is the best website to find a therapist in Seattle, WA. Browse profiles of local therapists or search by zip code. Find phone numbers or send an ...
Marriage counselor Seattle, WA - Yelp
www.yelp.com/search?find...marriage+counselor...Seattle%2C... Yelp
Reviews on Marriage counselor in Seattle, WA Sheri Vernon MA LMHC, The Gottman Institute, Christine Wylde, MA LMHC, Clarity Counseling Seattle, Connie ...
Map for marriage counseling seattle
Ads
Divorcing a Narcissist?
www.duboislaw.net/
It's not easy. We can help.
Call for a Consult Today.
Counselor in Seattle
www.stephaniemcounseling.com/
Individual and Couple Counseling
Relationship Counseling
Bellevue Counselor
www.jaimebaileylmft.com/
Relationship therapy for
individuals and couples.

Assassin's prioritized TODO manager

Priority Queue Operations

- **Insert(S, x)** inserts the element x into set S
- **Maximum(S)** returns the element of S with the maximum key
- **ExtractMax(S)** removes and returns the element of S with the maximum key
- *How could we implement these operations using a heap?*

Priority Queue Operations

- **Insert(S, x)**
  - Increment heap size and add x at the end
  - move the new element "upwards" (reverse-heapify)
  - O(log n)
- **Maximum(S)**
  - return S[1]
  - Time complexity?
- **ExtractMax(S)** removes and returns the element of S with the maximum key
  - save S[1], place S[heap_size(S)] in S[1]; Heapify(S,1)
  - Time?

Heap vs All (for Priority queues)

Data Structure	Pre-processing	Insert	Max	Extract Max
Linked List	**O(n)**	**O(1)**	O(n)	O(n)
Sorted Array	O(n logn)	O(n) (shifting)	**O(1)**	**O(1)**
Heap	**O(n)**	O(log n)	**O(1)**	O(log n)

Announcements

- Read through Chapter 6
  - Next class: Build heap, Heap Sort, Priority Queues
- HW2 due next Wednesday