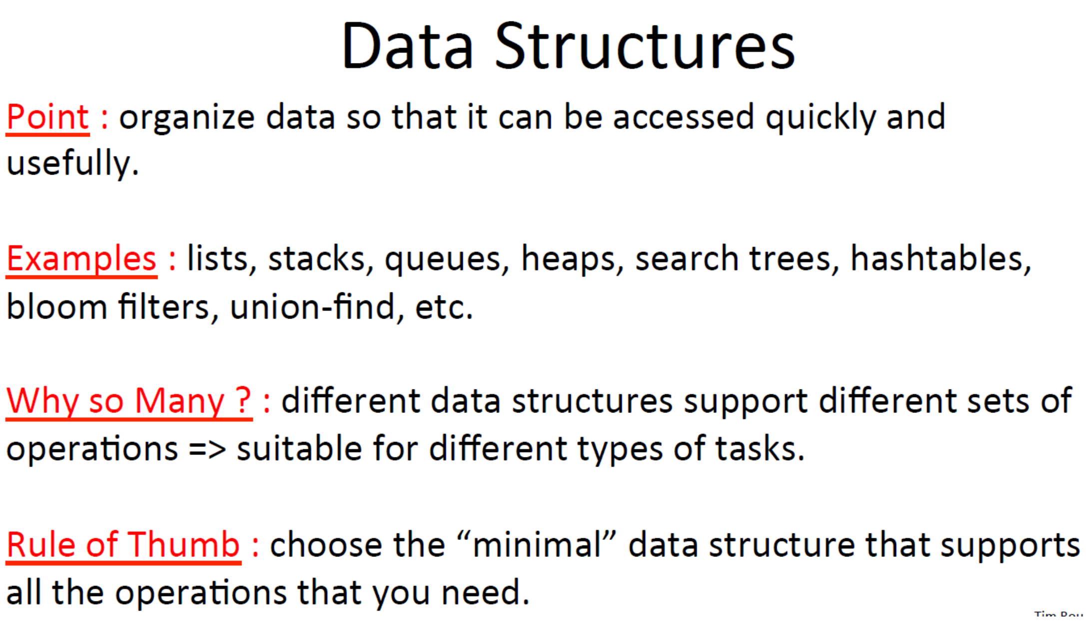
Part12-Heaps

Key videos	https://www.coursera.org/learn/algorithms-graphs-data-structures/lecture/iIzo8/heaps-operations-and-applications https://www.coursera.org/learn/algorithms-graphs-data-structures/lecture/KKqlm/heaps-implementation-details-advanced-optional
Note	Introduction to heaps and some of their applications, and some details about how they are typically implemented
Period	@2020/04/28 → 2020/05/06

Note

▼ Data Structures: Overview

- 基础介绍：针对不同问题使用不同的数据结构



- 使用数据结构的不同层次

Taking It To The Next Level

Level 0 - "what's a data structure ?"

Level 1 - cocktail party-level literacy

Level 2 - "this problem calls out for a heap"

Level 3 - "I only use data structures that I create myself"

▼ Heaps: Operations and Applications

- 支持操作：关键在Insert与ExtractMin都是O(logn)

Heap: Supported Operations

- A container for objects that have keys
- Employer records, network edges, events, etc.

Insert: add a new object to a heap.

Running time : O(log(n))

Equally well, EXTRACT MAX

Extract-Min: remove an object in heap with a minimum key value. [ties broken arbitrarily]

Running time : O(log n) [n = # of objects in heap]

Also : HEAPIFY (n batched Inserts in O(n) time), DELETE(O(log(n)) time)

- 应用1：HeapSort——转化为堆，然后n次ExtractMin操作

Application: Sorting

Canonical use of heap : fast way to do repeated minimum computations.

Example : SelectionSort ~$\theta(n)$ linear scans, $\theta(n^2)$ runtime on array of length n

Heap Sort : 1.) insert all n array elements into a heap
2.) Extract-Min to pluck out elements in sorted order

Running Time = 2n heap operations = O(nlog(n)) time.
=> optimal for a "comparison-based" sorting algorithm!

- 应用2：Median Maintenance——构造两个堆，然后结合ExtractMin操作，同时构造出ExtractMax操作，即可

Application: Median Maintenence

I give you : a sequence x1,…,xn of numbers, one-by-one.

You tell me : at each time step i, the median of {x1,….,xi}.

Constraint : use O(log(i)) time at each step i.

Solution : maintain heaps H_{Low} : supports Extract Max
H_{High} : supports Extract Min

Key Idea : maintain invariant that ~ i/2 smallest (largest) elements in H_{Low} (H_{High})

You Check : 1.) can maintain invariant with O(log(i)) work
2.) given invariant, can compute median in O(log(i)) work

- 应用3：Speeding Up Dijkstra

Dijkstra's Shortest-Path Algorithm

vertices # edges

-Naïve implementation => runtime = $\theta(nm)$
- with heaps => runtime = O(m log(n))

loop iteratios

Work per iteration [linear scan through edges for minimum computation]

- 其他应用：Event Manager

  "Priority Queue" – synonym for a heap.

  Example : simulation (e.g., for a video game)

  -Objects = event records [Action/update to occur at given time in the future]

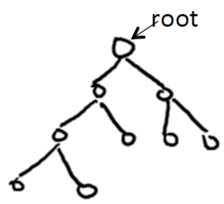
  - Key = time event scheduled to occur
  - Extract-Min => yields the next scheduled event

▼ Heaps: Implementation Details [Advanced - Optional]

- 堆的性质：
  - （树的实现）永远是平衡二叉树 balanced binary tree，会尽可能填满每一层

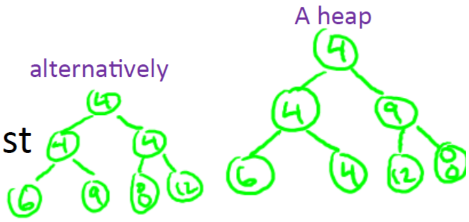
    ## The Heap Property

    Conceptually : think of a heap as a tree.
    -rooted, binary, as complete as possible

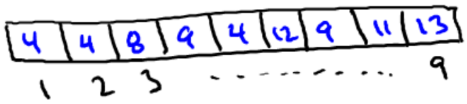
    Heap Property: at every node x,
    Key[x] <= all keys of x's children

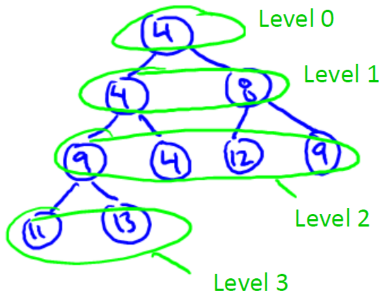
    Consequence : object at root must have minimum key value

  - （列表的实现）：父节点出现在[i/2]，子节点出现在2i与2i+1

Array Implementation





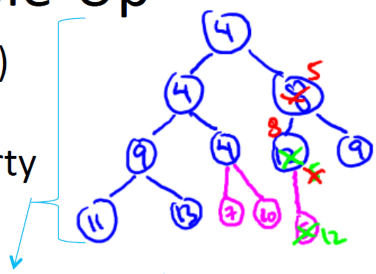
Note : parent(i) = i/2 if i even
= [i/2] if i odd

i.e., round down

and children of i are 2i, 2i+1

- 相关操作的实现
  - Insert and bubble up：如果出现打破堆的属性，然至少会出现O(logn)次的翻转，重新恢复性质

Insert and Bubble-Up



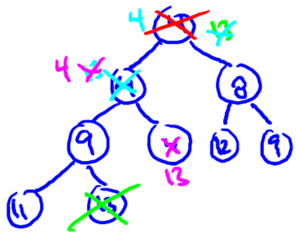
Implementation of Insert (given key k)
Step 1: stick k at end of last level.
Step 2 : Bubble-Up k until heap property is restored (i.e., key of k's parent Is <= k)

~ log2n levels (n = # of items in heap)

Check : 1.) bubbling up process must stop, with heap property restored
2.) runtime = O(log(n))

  - ExtractMin and bubble down：取出root，然后将最新的leaf上移，然后一步步与每层中较小的root进行swap直至结构恢复

Extract-Min and Bubble-Down



Implementation of Extract-Min

1. Delete root
2. Move last leaf to be new root.
3. Iteratively Bubble-Down until heap property has been restored

   [always swap with smaller child!]

Check : 1.) only Bubble-Down once per level, halt with a heap

2.) run time = O(log(n))

Reference

- Heaps 堆

堆積

堆（英語：Heap）是 计算机科学中的一種特別的樹狀 数据结构。若是滿足以下特性，即可稱為堆積：「給定堆積中任意 節點P和C，若P是C的母節點，那麼P的值會小於等於（或大於等於）C的值」。若母節點的值恆 小於等於子節點的值，此堆積稱為 最小堆積（min heap）；反之，若母節點的值恆 大於

W https://zh.wikipedia.org/wiki/%E5%A0%86%E7%A9%8D

Heap (data structure)

In computer science, a heap is a specialized tree-based data structure which is essentially an almost complete tree that satisfies the heap property: in a max heap, for

W https://en.wikipedia.org/wiki/Heap_(data_structure)