

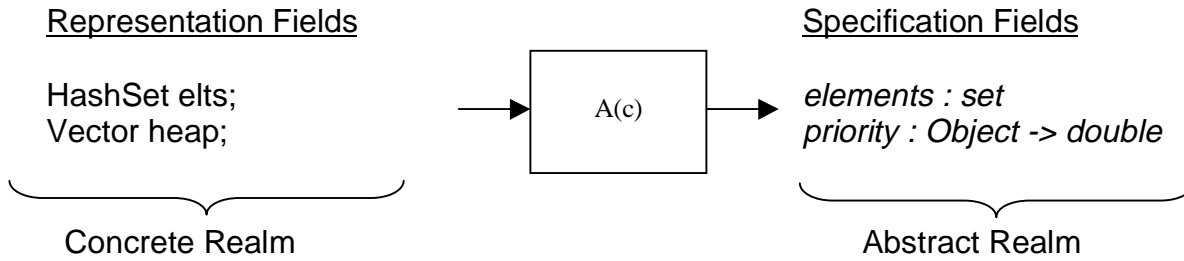
Recitation 3 : Additional Diagrams

The real content of this handout begins on the next page. This cover page only notes some discrepancies between this handout and the "main" R3 handout:

In this handout:

- 1) The rep invariant for the PQ has these additional items
 - a) `c.heap` is not null
 - b) `c.elts` is not null
 - c) `size(c.heap) >= 1`
- 2) The rep invariant and abstraction function have been re-written to be more consistent with lectures and lecture notes, instead of exactly like the R3 document.
- 3) The objects in the PQ have been re-numbered so that the dummy `heap.elements[0]` is called 'd', and the first object in the heap is called 'o1'. Therefore, all of the oN numbers are off-by-1 compared to with R3.

Priority Queue



Concrete Realm

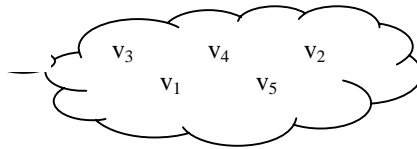
Notation:

HashSet elts : $\{v_1, v_2, v_3, v_4, v_5, \dots\}$ // $\{ \}$ notation for a set

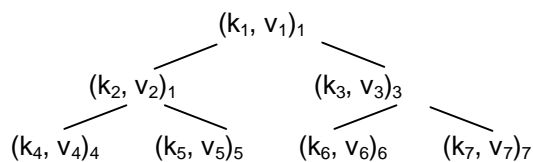
Vector heap : $[d, (k_1, v_1), (k_2, v_2), (k_3, v_3), \dots]$ // $[]$ notation for a list; (k, v) is an Entry

"Visualize" the rep:

HashSet elts :



Vector heap :



Rep Invariant

$R(c) =$

$c.\text{heap}$ is not null	$\&\&$ $c.\text{elts}$ is not null	
$\&\&$ $\text{size}(c.\text{heap}) \geq 1$		
$\&\&$ all i :	$c.\text{heap.elements}[i]$ is of type Entry	$1 \leq i < \text{size}(c.\text{heap})$
$\&\&$ all i, j :	$c.\text{heap.elements}[i].\text{value} \neq c.\text{heap.elements}[j].\text{value}$	$1 \leq i < j < \text{size}(c.\text{heap})$
$\&\&$ all i :	$c.\text{heap.elements}[i].\text{key} \leq c.\text{heap.elements}[2i].\text{key}$	$1 < 2i < \text{size}(c.\text{heap})$
$\&\&$ all i :	$c.\text{heap.elements}[i].\text{key} \leq c.\text{heap.elements}[2i+1].\text{key}$	$1 < 2i+1 < \text{size}(c.\text{heap})$
$\&\&$ all i, e :	$(e \text{ in } c.\text{elts}) \Leftrightarrow (\exists i, e = c.\text{heap.elements}[i].\text{value})$	$1 \leq i < \text{size}(c.\text{heap})$

In english:

- AND The heap and elts fields are never null
- AND The heap always has at least one element (a dummy)
- AND All elements of the heap except the first are of type Entry
- AND The value fields in the heap hold distinct values
- AND The key at i is smaller than the keys at $2i$ and $2i+1$, if they exist
- AND The set of elts is exactly the set of value fields in the heap

Abstraction Function

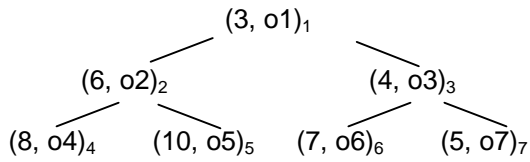
$q.\text{elements} = c.\text{elts.elements}$
 $q.\text{priority} = (c.\text{heap.elements}[i].\text{value} \rightarrow c.\text{heap.elements}[i].\text{key}) \mid 1 \leq i < \text{size}(c.\text{heap})$

Tracing a call to `pq.extractMin()`... for each step, decide if RI is satisfied.

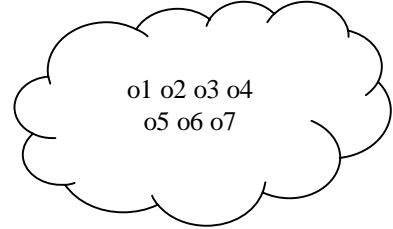
Vector heap;	HashSet elts;
--------------	---------------

... upon entry to `extractMin()`

[d, (3, o1), (6, o2), (4, o3), (8, o4), (10, o5), (7, o6), (5, o7)]

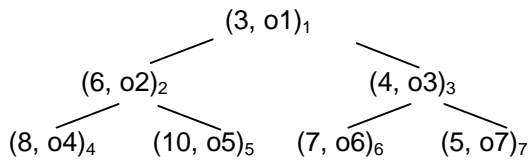


{o1 o2 o3 o4 o5 o6 o7}

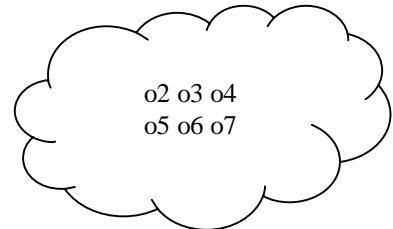


... after `elts.remove(min)`

[d, (3, o1), (6, o2), (4, o3), (8, o4), (10, o5), (7, o6), (5, o7)]

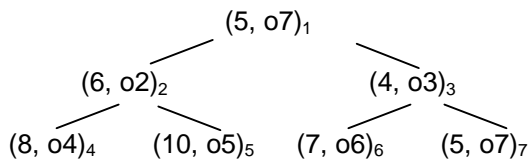


{o2 o3 o4 o5 o6 o7}

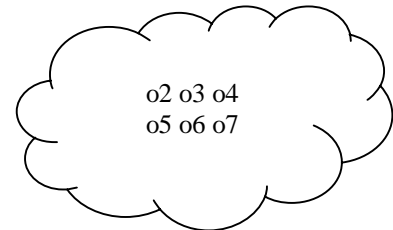


... after `heap.setElementAt(...)`

[d, (5, o7), (6, o2), (4, o3), (8, o4), (10, o5), (7, o6), (5, o7)]

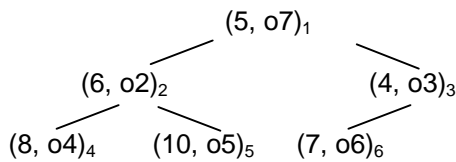


{o2 o3 o4 o5 o6 o7}

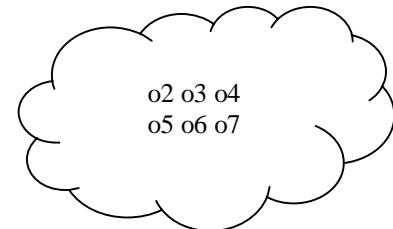


... after `heap.removeElementAt(...)`

[d, (5, o7), (6, o2), (4, o3), (8, o4), (10, o5), (7, o6)]

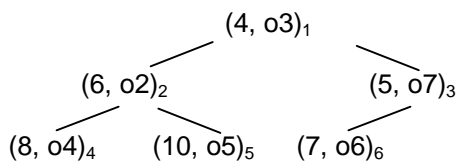


{o2 o3 o4 o5 o6 o7}

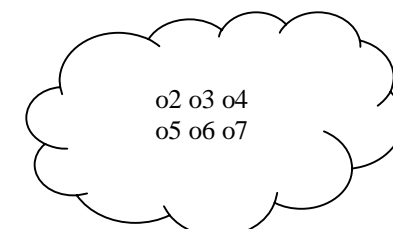


... just before `heapify(1)` calls `heapify(3)`

[d, (4, o3), (6, o2), (5, o7), (8, o4), (10, o5), (7, o6)]



{o2 o3 o4 o5 o6 o7}



... `heapify(3)` has no effect; everything is done and `o1` is returned.