Problem set 1

Theodore S. Norvell

6892

For both questions, follow the example in the notes.

Q0

(a) Design a syntactic and semantic interface for an unbounded queue. Include an abstract state, an abstract invariant (if needed), signatures for all methods, an initial state, and pre- and postconditions for all methods in terms of the abstract state.

(b) Design a concrete state and a linking invariant. (Use the dynamic array if you want.) Draw a picture of the linking invariant. [Hint: By splitting the queue over 2 dynamic arrays you can arrive at a very efficient implementation.]

(c) Supply an implementation for each method, and for initialization, in terms of the concrete state.

Q1 A union-find structure represents a partition of a set $\{0, ..n\}$ into a set of nonempty subsets S such that each element of$\{0, ..n\}$ is a member of one (and only one) element of S. For example $\{\{0, 2, 4\}, \{1, 3, 5\}\}$ is a partition of $\{0, ..6\}$ as are $\{\{0, 1, 2, 3, 4, 5\}\}$ and $\{\{0\}, \{1\}, \{2\}, \{3\}, \{4\}, \{5\}\}$. The following notation may be useful: Given a number i and a partition S, $[i]_S$ is the equivalence class of i, i.e. that element of S that contains i.

The initial state of a union-find object has each element of $\{0, ..n\}$ in its own subset (i.e., the size of S is n). There are two operations on the union find structure:

- find takes two number as arguments and returns true if they are in the same element of S, and false if they are in different elements of S
- union takes two numbers as arguments and combines the elements of S that they are in. Thus if find(i, j) is false, union(i, j) will reduce the size of S by 1. For example if the initial value of S is $\{\{0\}, \{1\}, \{2\}, \{3\}, \{4\}, \{5\}\}$ then, after union$(2, 3)$ is executed, the value of S will be $\{\{0\}, \{1\}, \{2, 3\}, \{4\}, \{5\}\}$.

(a) Design the syntactic and semantic interface for a union-find class. Include an abstract state, an abstract invariant (if needed), signatures for all methods, an initial state, and pre- and postconditions for all methods in terms of the abstract state.

(b) A well known data structure for union-find objects is a set of trees so that the elements of each tree are the elements of one element of the partition. Each tree can be represented by simply be linking each node to its parent. Thus the whole set of trees can be represented by an array of numbers p so that the parent of i is $p(i)$. If i is a root, then $p(i)$ has some special value to indicate that i is a root. Design a concrete state and a linking invariant. You may find that the linking invariant is a bit tricky to state mathematically. In that case, just use clear English.

(c) Supply an implementation for each method, and for initialization, in terms of the concrete state.