Part 8
—
Breadth-First Search

CSCI 3110 Code

Summer 2015

Next we implement BFS. Here's the type signature of the function we want:

```
bfs :: AdjList v vl el → Forest V E
```

Again, we implement *bfs* using *Algos.Graphs.Traversal.traverse* combined with the right vertex set data struture, a queue this time:

```
bfs = traverse makeVertexQueue
```

A vertex queue is of easy to implement using the queue implementation from *Algos.DS.Queue* stored in an *STRef* combined with an array of explored vertices:

```
data VertexQueue s = VertexQueue (STArray s Int Bool) (STRef s (Queue (V, [(E, V)])))
```

To create such a vertex queue, we simply allocate a new Boolean array of size n all of whose entries are initially *False*—all vertices are initially unexplored—and we create a new *STRef* initially storing an empty queue:

```
makeVertexQueue   :: Int → ST s (VertexQueue s)
makeVertexQueue n = VertexQueue <$> newArray (1, n) False <*> newSTRef emptyQueue
```

Next the implementations of the two set operations:

```
instance VertexSet VertexQueue where
  add (VertexQueue _ qu) v p = modifySTRef qu (flip enqueue (v, p))
```

```
remove (VertexQueue exp qu) = readSTRef qu ≫= rem
  where rem q = case front q of
                  Nothing        → writeSTRef qu q ≫ return Nothing
                  Just p@(v, _) → do e ← readArray exp (vIx v)
                                      if e then rem (dequeue q)
                                        else  do writeSTRef qu (dequeue q)
                                                 writeArray exp (vIx v) True
                                                 return (Just p)
```

add simply enqueues the given pair (v, p). *remove* reads queue and passes it to the helper function *rem*. If the given queue is empty, we write this information back into the *STRef* and return *Nothing*. Otherwise, we inspect the front pair p. If its vertex v is already explored, which we check by reading the array *exp*, then p should not be returned, so we recurse on the tail of the queue using *rem* (*dequeue q*). Otherwise, we store the tail as the new queue content, mark v as explored, and finally return *Just p*.