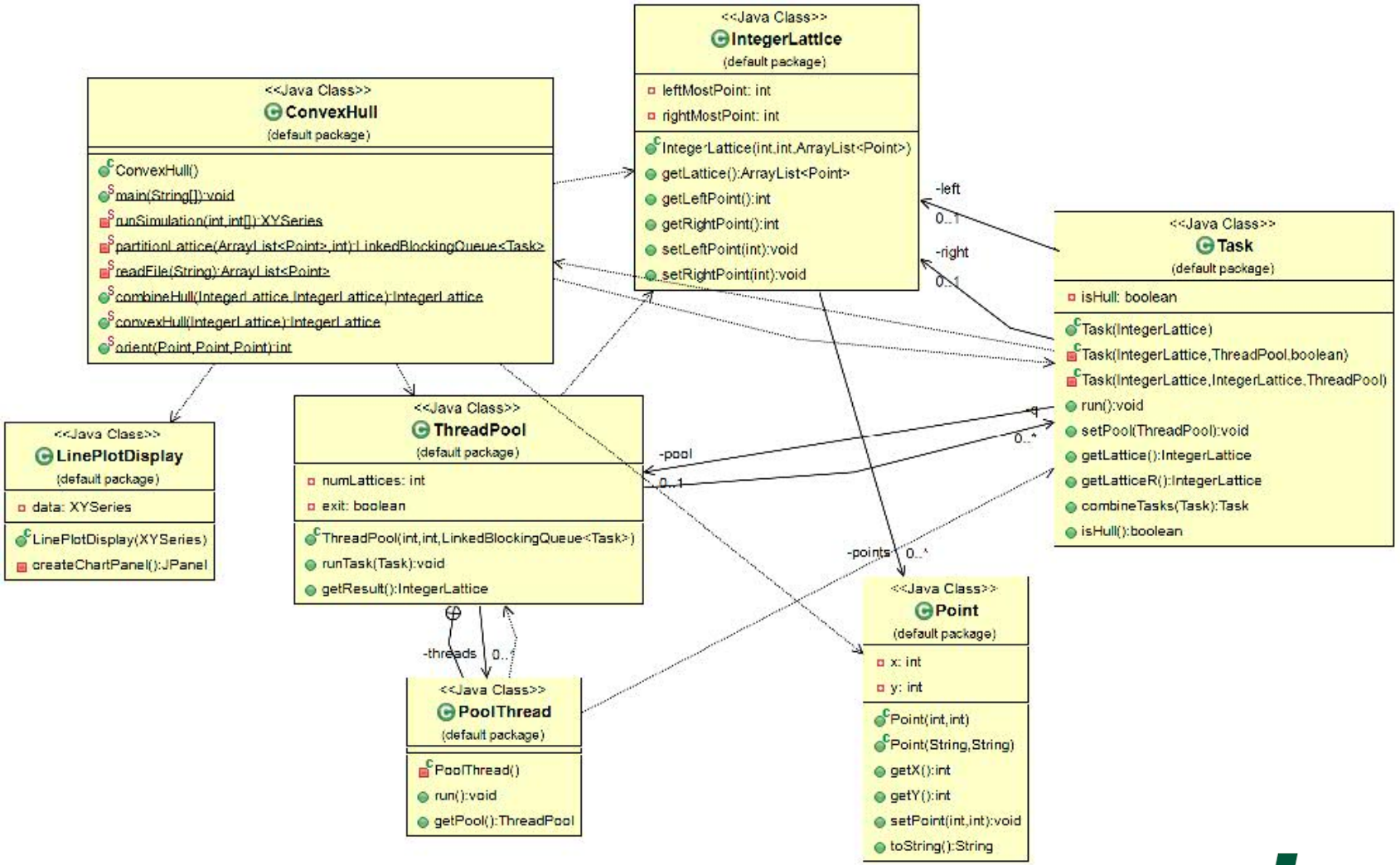
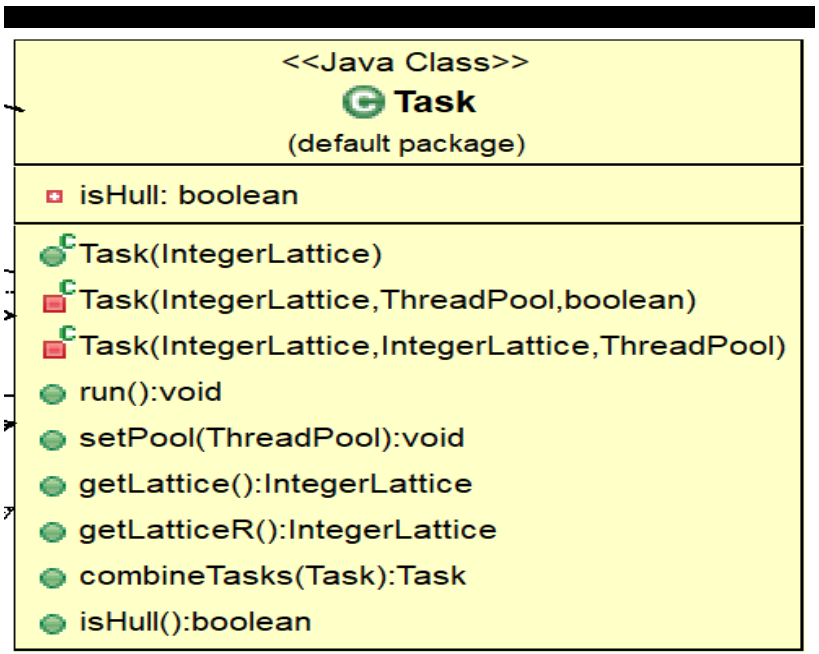
Concurrent Convex Hull
CPSC 222

University of Northern British Columbia

Solution Architecture



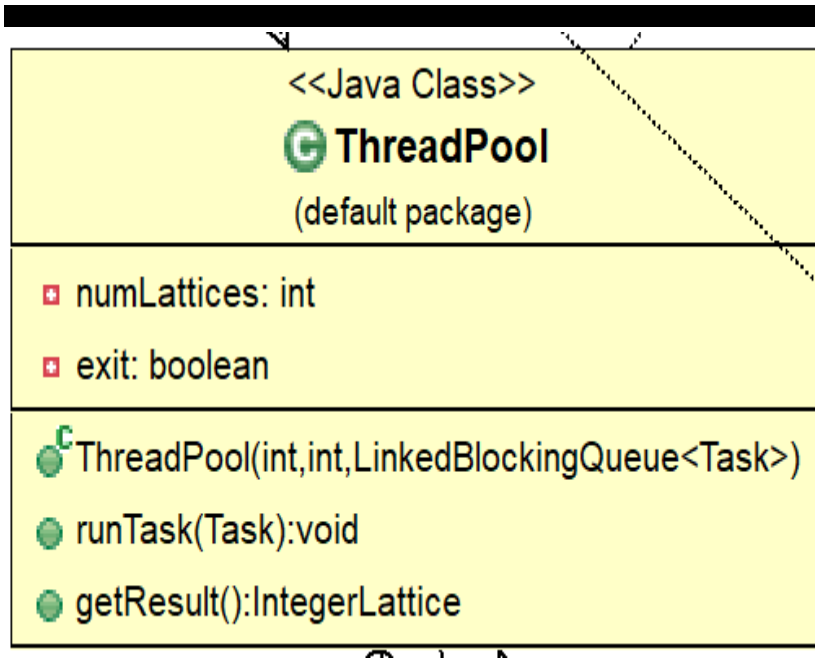
Solution Architecture: Tasks



Components:

1. Unit of work: Merge/Hull
2. Merge tasks have a left and right lattice
3. You can combine two tasks
4. Each task knows if it represents a convex hull

Solution Architecture: Tasks



Components:

1. Responsible for concurrency aspects
2. Most work done in the `runTask()` method
3. `LinkedBlockingQueue` used coordinate tasks
4. Boolean `exit` flag used for program termination

run()


```
public void run() {

    while (!exit) {
      Task task = null;
      synchronized (q) {
        while (!exit && q.isEmpty()) {

            q.wait(); {...}
        }

        if (q.peek().isHull()) {
          while (!exit && q.size() < 2) {
            try {
              q.wait();
            } catch (InterruptedException e) {
              System.out.println(e.getMessage());
            }
          }


          task = q.poll();

          if (!exit) {
            if (q.peek().isHull()) {
              task = task.combineTasks(q.poll());
              numLattices--;
            } else {
              q.add(task);
              task = q.poll();
              task.setPool(getPool());
            }
          } else {
            q.add(task);
          }

        } else {
          task = q.poll();
          task.setPool(getPool());
        }


      } // end of synchronization
```

UNBC

```
            if (!exit && task != null) {
              task.run();
            }


          if (numLattices == 1) {
            exit = true;

            synchronized (q) {
              q.notifyAll();
            }

          }

        }

      }
```

Solution Specifications

1. The Jarvis March algorithm was used for convex hulls $\mathcal{O}(nh)$
2. Merging two convex hulls done through an adaptation of the Jarvis algorithm $\mathcal{O}(n)$
3. I currently have a stupid design of where I do not take advantage of OOP within Tasks.
4. Largest challenge was avoiding deadlock with merging tasks and finding exit condition.