Counterintelligence

November 26, 2016

- *Likely RELATIONSHIP TO FINAL EXAM:* Debugging algorithms by identifying key properties and producing small examples that push on their weaknesses is a key ability. We anticipate asking about these ideas on the exam. We do **not** anticipate asking about the Olympic Scheduling Problem on the exam.

Recall the Olympic Scheduling Problem. The key features of this problem were:

- The input is a list of events with start time, finish time, and value. Assume that all are positive and that each event's finish time is after its start time.
- The solution is the best set of non-conflicting events.
  - Two events conflict if each one starts **before** the other finishes (i.e., they overlap in time).
  - The best solution is the one with the single highest-valued event, breaking ties by comparing next highest-valued events (where both solutions are assumed to have as many 0-valued events as needed to break all ties).

Consider the following algorithm that attempts to solve the problem greedily by considering the events in order of finish time and adding any event that does not conflict with a higher-valued event:

```
ValueIncreasingSoln(E):
  sort E by increasing finish time    // in O(|E| lg |E|)

  result = new empty list of events   // the result so far
  current = no event                  // the current event under consideration

  for each e in E:
    if there is no current event:
      // just the first time through the loop
      current = e
    else if start(e) >= finish(current):
      // we've passed the range current conflicts with
      add current to result
      current = e
    else if value(e) > value(current):
      // we've hit a higher-valued conflicting event
      current = e
    else:
      // otherwise, e and current conflict, but current is higher-valued
      // we do nothing and ignore e

  // polish off the last current event
```

```
  if there is a current event:
    add current to result

  return result
```

This seems promising. Let's investigate.

1. Sketch the key points in a (brief!) proof that the optimal solution must include any event that conflicts only with lower-valued events.

2. Despite this promising result, the greedy algorithm is **not** correct. Give a small counterexample on which this greedy approach fails. Be sure to clearly indicate both what the greedy approach produces and what the optimal solution is.