CME 212 Feedback hw0

Grading TA: amyshoe@stanford.edu
January 29, 2017

File: Graph.hpp
Line: 108

This implementation doesn't satisfy the specs! Two nodes from different graphs that have the same point in their respective graphs would here be returned as equal nodes, when they're not even part of the same graph Also the specs specifically say that equal nodes have the same *index*, no necessarily the same point. In fact, our specs overall allo for multiple (distinct) nodes that have the same point. These should be considered different nodes.

```
/** Test whether this node and @a n are equal.
     *
     * Equal nodes have the same graph and the same index.
     */
    bool operator==(const Node& n) const {
      // HW0: YOUR CODE HERE
      return norm_1(graph_->node_list[uid_]) == norm_1(n.graph_->node_list[n.uid_]);
    }
```

File: Graph.hpp
Line: 134

your global ordering here is *technically* correct, but only because your == op is incorrect. Once you fix op==, this < implementation will no longer obey trichotomy (two nodes x and y with same index but different graphs will not return true for *any* of x==y, x<y, y<x). Also, you'll have some weirdness of having op== be implemented iwth index and < with points. Rethink this function before the next assignment! Come to OH if you want to chat in more detail!

```
bool operator<(const Node& n) const {
      // HW0: YOUR CODE HERE
      return norm_1(graph_->node_list[uid_]) < norm_1(n.graph_->node_list[n.uid_]);
    }
```

File: Graph.hpp
Line: 246

In op< you do a good job of testing all posibilities of first/second uid being min/max, but here, you'll return that edge (a ,b) !== (b,a), which violates the spec.

```
return graph_->edge_list[uid_].first == e.graph_->edge_list[e.uid_].first ↩
    &&
        graph_->edge_list[uid_].second == e.graph_->edge_list[e.uid_].↩
            second;
      }
    }
```

File: Graph.hpp
Line: 339

Duplicate code. Why not just use the has_edge() function?

```
for (size_type i = 0; i < edge_list.size(); ++i) {
        const auto &temp = edge_list[i];
        if ((a.uid_ == temp.first && b.uid_ == temp.second)
         || (a.uid_ == temp.second && b.uid_ == temp.first)) {
```

File: Graph.hpp
Line: 347

By having only an edge id in your Edge class and not two node ids, you're unable to meet the specification in this function that edge.node1()==a and edge.node2()==b In order to meet all the specs, you should consider another implementation! (You'll probably have to end up storing two node ids. But if you think of a more creative solution, let me know! I'd be really curious to hear it!)

```
return edge(i);
```

Your hw0 grade:

$\boxed{8}$

Grade	Explanation
0	Did not try, did not hand in, or submitted too late with no late-days left.
1-2	Poor. Little to no serious attempt on this homework. Submission has barely changed since last homework (if any) or did not follow the guidelines at all.
3-4	Poor. Did not finish but a good attempt. Conveyed the message of understanding the material.
5-6	Fair. Code is buggy but could be debugged and/or some major conceptual errors. Code does work and produces output along homework guidelines.
7-8	Good. Code compiles and runs properly with mostly the right output. Some mistakes and minor conceptual errors that could be worked on.
9-10	Excellent. No or very few minor mistakes. Conveyed solid understanding of the material.
11	Exceptional. Showed extra insight. Implemented features that improved the code beyond what was requested.