Ingeborg Ødegård Oftedal, Paul Philip Mitchell, Hallvard Jore Christensen and Kyrre Laugerud Moe

Design document: HW5

By wrapping calls done to Space in a SpaceProxy we are able to add the option of doing all calls to Space asynchronous. This means that network calls can be made will the Computer is working on other tasks. This mechanic also allows for a best effort update of the global lower bound. Therefore there there need not be a specific structure to where and when the developer chooses to update the lower bound. SpaceProxy is also implemented as a singleton. This means that whenever it is wished to update the lower bound, a single call can be made to SpaceProxy, and the value is updated.

The specific method to update the lower bound through SpaceProxy is updateGlobal(). This call tells SpaceProxy to update Space's global value. Space has it's own synchronized updateGlobal(). This method checks the value of the new global against the current. If the new global is considered better, Space's global is updated. Space also makes sure that the current global is piggybacked on closures as they are taken from the readyStack.

To summarize, the method for updating the shared (global) value is as follows. When a completed tour has been found, the tour length is wrapped in a Global object, and sent asynchronously to Space. Space checks it against its current global, and updates it if necessary. All closures receives the current global right before they are sent to a computer.

Pruning is done after two conditions. First a cheap check is done comparing the partial path against the current global. If it is already longer, the tree can be pruned. Otherwise a more thorough check is run. Here the length of the partial path and the lower bound are added and compared with the global. If it is longer than the global, we can be sure it is not part of the solution, and the tree can be pruned.

From the last assignment we have changed the ready queue to a LinkedBlockingDeque which functions as a blocking stack. This effectively makes the system depth first search. We would have liked to make it a priority queue, but were unable because of time concerns. This would have made the system perform best first search.