**Before:**

rheaParams.budget\_type = Constants.ITERATION\_BUDGET;

rheaParams.iteration\_budget = 200;  
rheaParams.individual\_length = 12;  
rheaParams.heurisic\_type = Constants.CUSTOM\_HEURISTIC;  
rheaParams.mutation\_rate = 0.5;

**After:**

rheaParams1.budget\_type = Constants.ITERATION\_BUDGET;  
rheaParams1.iteration\_budget = 200;  
rheaParams1.individual\_length = 10; // reduced from 12  
rheaParams1.heurisic\_type = Constants.ADVANCED\_HEURISTIC; // changed from CUSTOM\_HEURISTIC  
rheaParams1.mutation\_rate = 0.5;  
rheaParams1.mcts\_depth = 14; // increased from 12

**Results:**  
{ 1 10 5 −1 4 4 6 6 } [RHEA, RHEA, RHEA\_modified, RHEA\_modified]  
  
N  Win  Tie  Loss  Player (overtime average)

50 42.0% 0.0% 58.0% players.rhea.RHEAPlayer (1.0)

50 58.0% 0.0% 42.0% players.rhea.RHEAPlayer (0.0)

50 42.0% 0.0% 58.0% players.rhea.RHEAPlayer (2.0)

50 58.0% 0.0% 42.0% players.rhea.RHEAPlayer (0.0)

**Conclusions:**

1. Decreased `individual\_length` to encourage competitiveness.
2. Increased `mcts\_depth` to encourage the exploration of more actions down the tree.
3. Implemented [ADVANCED\_HEURISTIC](https://github.com/GAIGResearch/java-pommerman/wiki/Heuristics#advanced-heuristic).