Katholieke
Universiteit
Leuven

**Department of
Computer Science**

PROJECT

Genetic Algorithms and Evolutionary Computing
(B-KUL-H02D1A)

Verbois-Halilovic

Sten Verbois (??)
Armin Halilovic (r0679689)

Academic year 2017-2018

Contents

Introduction

Tasks

Task 2: Initial experiments

Task 3: Stopping criterion

To implement a new stopping criterion, we looked at the commonly used termination conditions outlined by the book. There we see the following suggestions:

1. Maximally allowed CPU time elapses.
2. Total number of fitness evaluations reaches limit.
3. Fitness improvement remains under threshold for a given period of time.
4. Population diverisity drops under threshold.

The first and second citerion are useful, either to guarantee the evaluations do not go on forever, or when there is some kind of constraint on system resource usage. In the project template, we already have the guarantee of eventual termination because of the limit on the number of generations, and we do not have to account for system resource constraints.

The fourth criterion is also already present in the template and can be adjusted via the GUI. The default value is so strict (95% equal individuals), it practically is never reached.

So we decided to implement the fourth criterion. Termination occurs when the fitness of the best individual does not improve above a treshold for a given period of time. This period of time is expressed in terms of a certain number of generations. We chose to define this number of generations to be a percentage of the specified maximum number of generations.

Task 4: Path representation

Task 5: Local optimisation

Task 6: Benchmark problems

Task 7: Optional tasks

7a: Parent selection

7b: Survivor selection

7c: Diversity preservation

Conclusion

Weak points

There are a couple of weak points in our report:

- The speed of our Sudoku implementations in CHR are not that great. We did try to improve it with the heuristics, but we feel it is still too slow. We think this was more due to the fact that we were still novices with how CHR worked as our Hashiwokakero CHR implementation is quite fast.
- The implementation of the connectivity constraint in ECLiPSe for Hashiwokakero is also a rather weak point. Ideally we should have implemented it using disjoint sets like in CHR but we couldn't figure out how to do this correctly.
- We think that implementing a Hashiwokakero solver using a graph representation is a more logical approach than the one we have used, but we had problems with ECLiPSe and how to express the no crossing bridges constraint.

Strong points

The strong points in our report are:

- Our Hashiwokakero CHR implementation is quite fast thanks to our improvements set.
- We think we have good heuristics for Sudoku in CHR. Certainly the heuristic for the alternative viewpoint has made good improvements on the speed of the search.

Lessons learned

We have learned a lot of things during this project due to the fact we often had to re-track our steps. If we would now have to do an other project with these systems we think we would have a better idea for how to start and what to think about. We spent quite some time with both ECLiPSe and CHR so we now have a much better idea of what the strong points are and the weak points are for each system.

Appendix

We started working on this project before the Easter holiday. In the beginning we often lost quite some time, since we didn't really know how the systems worked. During the second week of the Easter holiday, we continued to work on the project each evening and we finished the Sudoku task and the ECLiPSe part of hashiwokakero before the end of the holiday. We then had to halt our work for a while since we had a deadline for a ridiculously large project for another course. As the semester was coming to an end other deadlines and an exam were coming up so we had to manage those first. Thus it was only at the start of the study period that we could continue working. From the start of the study period we tried to spend around 6 hours of work each day for this project. The work was not really divided since we were doing pair programming most of the time. Sometimes someone made individual changes when they had time but most of our work was done online using Hangouts and its screen sharing functionality. We could argue that by doing pair programming we lost quite some time, which is true, but by doing this we worked very closely together and we learned quite a lot. We have each individually put more than 100 hours in this project.