LEHRSTUHL FÜR INFORMATIK II
RWTH Aachen · D-52056 Aachen · GERMANY
http://moves.rwth-aachen.de/

Prof. Dr. Ir. J.-P. Katoen
Harold Bruintjes, Christian Dehnert, Sauvik Bhattacharya

Software lab summer term 2014
Implementation of Heuristic Algorithms for Board Games

– Assignment 1 –

Next meeting is on 25.04.2013. Upload your code and report before the meeting.

Task 1

Create three different playing fields according to the file format specification[1]. The fields should be no larger than 50 by 50 cells. Please upload your ASCII text files until **Friday, 18.04.2014** to your repository.

Task 2

Devise and implement a data structure that represents a playing field. Your design should be modular and allow for maintenance and subsequent changes but pay attention to performance, too. The maximal field size will be 50 by 50.

Furthermore implement a routine that reads in arbitrary (but valid) fields and represents them using your data structure.

Task 3

Devise an algorithm which decides if a given move is valid according to the specification. Consider possible transitions and overwrite stones. Bombs and special fields will be dealt with later.

Implement your algorithm using the data structures from task 2. Your program should decide if any given move is valid and if it is compute the successor state and output it (e.g. print to stdout).

Task 4

Invent two non-trivial[2] and useful functions that rate the result of any given move. One function will rate during ‘normal’ gameplay (when placing stones), another rates the bomb moves. The function’s return value should represent the rating of the resulting state for an arbitrary player. You need not implement these functions yet.

Do not forget to elaborate on your heuristics in the report and argue why you think they are the best among all that you have considered.

Task 5

Extend your program from task 3 so that successor states are correctly determined also on maps with special fields (choice, inversion, bonus and expansion fields). Additionally take into account overwrite

[1]The specification can be found in the shared repository (see remarks).
[2]This excludes simply counting the number of your own tiles.

stones. Remember that choice fields require an additional parameter for the chosen colour and bonus fields require information on the chosen bonus stone.

Test your program with suitable maps.

Submission of solutions:

You will find all necessary materials like specifications and a LaTeX template in the following (read-only) repository:

`https://sselab.de/lab9/private/git/swp-i2-2014-pm`

Please upload your code and your report (tex file) to your group repository. Each group has a private repository

`https://sselab.de/lab9/private/git/swp-i2-2014-gX`

where X is the group number. Please provide a *consistent* layout of the repository where you submit your files, and tell us where we can find them. Best practise is to provide us with a 'stable' branch that we can use.

Should you repeatedly miss the deadline (i.e. try to deliver anything after the group meeting) or submit faulty solution you can be expelled from the lab. There will also be hard deadlines which require that certain parts of your code must work in order to continue the lab.

For further questions and issues please use the mailing list `swp-i2-2014-pm@lab9.sselab.de` which can be used by all participants and assistants. Everybody should take the opportunity to answers others' questions whenever possible.