Software Praktikum SS 2016:
Implementing heuristic algorithms for board games

Betreuer: Harold Bruintjes, Stephan Kölker

Terms

- A **map** specifies the *board* to play on, and consists of various *tiles*.
- A **tile** (synonymous with *field*) is part of a map, and specifies a single place that a *player* can place *stones* on.
- A **transition** is a relation between tiles. If there is a transition between two tiles, they are said to be *neighbouring*.
- A **stone** is a piece the player places on a tile. Stones have colour which indicates to which player they belong to.
- **Recolouring** a stone means the stone will belong to a different player, namely the one whose colour it is recoloured to (synonymous with *capturing*).
- A **player** is an entity partaking in the game. A player has an assigned number and associated colour. Players can place a stone on the map during their turn.

Game map specifications

A map is represented in the following (text) format:

```
[# of players]
[# of overwrite stones]
[# of bombs] [Strength of bombs (radius)]
[Map height] [Map width]
```

`Map`

```
Transitions
```

⋮

where the following rules apply:

`# of players` $\in \{2, \ldots, 8\}$: gives number of players for the map.

`# of override stones` $\in \mathbb{N}$: gives the number of override stones. These stones can be placed on occupied tiles.

`# of bombs` $\in \mathbb{N}$: gives the number of bombs the players initially have. Bombs are used in the last stage of the game, and are used to clear areas of the map and their associated tokens.

`Strength of bombs (radius)` $\in \mathbb{N}$: gives the effect radius of bombs.

`Map height` $\in \{1, \ldots, 50\}$: gives the number of rows of the map.

`Map width` $\in \{1, \ldots, 50\}$: gives the number of columns of the map.

`Map` describes the properties/tiles of the map. Each tile is separated by a single space from the next tile. The elements are specified using $\{\texttt{0}, \ldots, \texttt{8}, -, \texttt{c}, \texttt{i}, \texttt{b}, \texttt{x}\}$, for which the following conditions hold:

- The number `0` means an empty tile.
- The numbers `1,...,8` mean a tile that is occupied by the respective player.
- $-$ indicates a "hole" in the map.
- `c` (choice) means a choice tile. **After** occupying such a tile, a player switches his stones with that of another player (possibly himself).
- `i` (inversion) means an inversion tile. **After** occupying such a tile, the stones of all players are switched, in such a way that the stones of player i are assigned to player $[(i \bmod n) + 1]$, where $1 \leq i \leq n$.
- `b` (bonus) means a bonus tile. **After** a player occupies such a tile, he has to choose to receive either a bomb or an override stone. The choice is added after the tile is occupied.
- `x` (expansion) means an expansion stone. A tile with an expansion stone is considered occupied, but the player is unspecified.

`Transitions` describe the transitions of the map. For a detailed explanation see the next section.

Remark: Keep in mind that map specifications can terminate a line with either \n or \r\n.
The following figures show an example map:

```
1   3
2   6
3   4 2
4   15 15
5   - - - - - 0 0 0 0 0 - - - - -
6   - - - - - 0 0 0 0 0 - - - - -
7   - - - - - 0 0 0 0 0 - - - - -
8   - - - - - 0 0 0 i 0 - - - - -
9   - - - - - 0 0 0 0 0 - - - - -
10  0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
11  0 c 0 0 0 0 1 2 3 0 i 0 0 0 0
12  0 0 0 0 0 0 3 1 2 0 0 0 0 0 0
13  0 0 0 b 0 0 2 3 1 0 0 0 0 0 0
14  0 0 0 0 0 0 0 0 0 0 0 0 0 b 0 0
15  - - - - - 0 0 x 0 0 - - - - -
16  - - - - - 0 x x x 0 - - - - -
17  - - - - - 0 0 x c 0 - - - - -
18  - - - - - 0 0 0 0 0 - - - - -
19  - - - - - 0 0 0 0 0 - - - - -
20  6 0 0 <-> 9 1 1
21  7 14 4 <-> 7 0 0
```

Figure 1: Example map

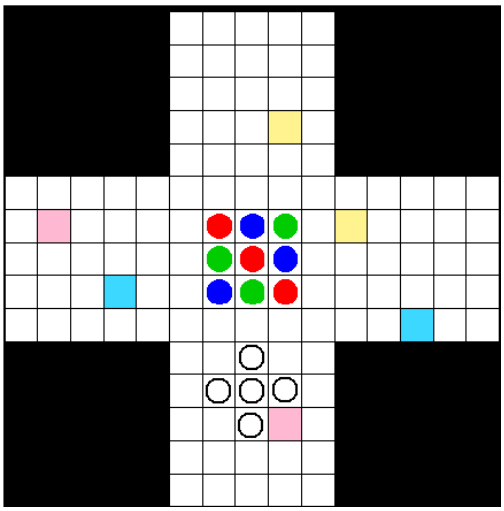

Figure 2: Graphical representation

The following table gives the parameters that apply for the map from Fig. 1 and Fig. 2:

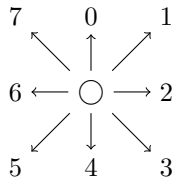

Figure 3: Possibilities for transition directions

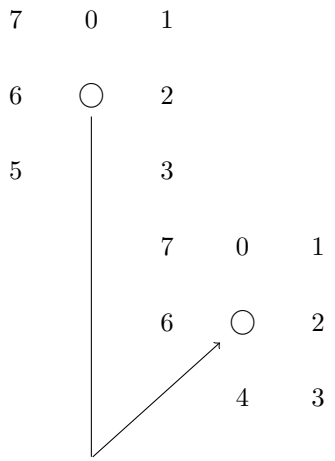

Figure 4: Example transition with possible directions

Section	Property	Value
Line 1:	`# of players`	3
Line 2:	`# of override stones`	6
Line 3:	`# of bombs`	4
Line 3:	`Strength of bombs (radius)`	2
Line 4:	`Map height`	15
Line 4:	`Map width`	15
Lines 5-19:	`Map`	see figure 2
Lines 20,21:	`Transitions`	see section about transitions

Transitions and directions

Transitions are the last part of a map definition. Using transitions the neighbouring relations of tiles can be extended. A transition definition uses the format $x_1 \sqcup y_1 \sqcup r_1 \sqcup$`<->`$\sqcup x_2 \sqcup y_2 \sqcup r_2$, where for $i \in \{1, 2\}$ the coordinates (x_i, y_i) indicate the position of the i^{th} tile and r_i indicate to what direction the transition applies. These directions are numbered according to Fig. 3. Figure 4 shows an example transition. If the top left tile in Fig. 4 has position $(0, 0)$, then the format of the transition would be as follows: `0 0 4 <-> 2 3 5` (or symmetrically: `2 3 5 <-> 0 0 4`). Every tile has at all times at most 8 neighbouring tiles, where at most one transition is allowed in each direction. When following a transitions across the map, the direction of the transition has to be taken into account. That is, a transition allows the direction of a path to be changed. **Important:** Transitions can only be extended: they can only be added to tiles that do not have a neighbouring tile in that direction already. This is the case for tiles which are adjacent to a hole in the map or which lie on the border of a map.

Gameplay

- The game is played with up to 8 players. Each player is assigned a number which determines the order of taking turns (in order of increasing numbers).
- A game consists of two phases: phase one, the *build phase*, and phase two, the *elimination phase*. In the build phase a player can use all possible stones (as long as allowed by the rules) except for bombs. This phase ends as soon as no player can make a move. The second phase immediately follows the first phase if any bombs are available. During the second phase, players can use their bombs to improve their situation against the other players on the map.
- Regular moves in phase 1 are:
  - Placing a stone on a valid free tile.
  - Using an override stone on a valid occupied tile or expansion stone.

  Regular moves in phase 2 are:
  - Placing a bomb on an existing tile of choice.
- **Phase 1 - build phase:** First player 1 is assigned the turn, and players take turns in order. During each turn, a player must make a move if possible. If a player cannot move he will be skipped.
- A move is allowed when a player encloses a row of one or more stones from other players, or expansion stones, in any direction, between an already placed stone of his colour and the newly placed stone. Here, transitions have to be taken into account.
- A stone can only be placed on a tile that contains no other stone and is not an expansion stone. However, an override stone can be used to place a stone on a tile that already has a stone, or is an expansion stone.
- The stones of an opposing player cannot be captured across the stones of the current player, including the newly placed stone.
- The recoloured stones must lie in a direct line (taking transitions into account) between the newly placed stone and some other stone of that player.
- All opposing stones that have been enclosed by the move of a player are coloured to the colour of the current player, even if this is disadvantageous to that player.
- If an expansion stone is overridden, the move is also allowed if no other stones are recoloured. (Expansion rule)
- After placing a stone on a special (choice, inversion, bonus) tile, this tile becomes a regular tile.
- Phase 1 ends, when none of the players can make a move.
- **Phase 2 - elimination phase:** In phase 2 only the dropping of bombs is allowed. Again players take turns, starting with the player after the last player that could not take a turn anymore.

- Bombs can be thrown anywhere, except for holes. They turn all tiles to holes that are reachable within n steps from the centre of the explosion, where n is the strength of the bombs as defined by the map. This means that transitions must be taken into account.
- **End of the game:** The game has ended, when no players can throw any more bombs. At that point, all the stones are counted, and players are ranked based on their number of stones in the map, where the player with the most stones wins. Disqualified players are ranked last.
- **Disqualification:** Invalid moves (or timeouts) cause immediate disqualification of the player making the move. A disqualified player is no longer part of the game and cannot take any more turns. However, the stones of the disqualified player that were already on the map remain on the map.

Further hints

- The repository must contain Makefiles/Antfiles which can successfully compile the AI.
- The tool should display some documentation describing the usage when invoked with `-h` or `--help` as an argument.
- The complete, **running** code has to be pushed in the git repository **before** the submission deadline. More information on git can be found under [1].
- Sockets can be used for network communications in both C++ and Java.
- Only the use of standard libraries is permitted. If anything is unclear please ask us.
- C++-Code has to compile using g++ 4.8 or clang 3.2. Java-code has to be compiled and run using Java 7.
- Details about SSELab can be found under [2].

References

[1] Git: `http://git-scm.com/documentation`

[2] SSELab: `http://sselab.de`