Software Praktikum SS 2016:
Implementing heuristic algorithms for board games

Betreuer: Harold Bruintjes, Stephan Kölker

Message format

Structure of a message. Unless specified otherwise, integers are unsigned and encoded in the network byte order.

Type (8-Bit-Integer)	Length of the message n (32-Bit-Integer)	Message data (n bytes)

The following messages have been defined (ordered by type number):

(1) Is used to send the **group number** of the client as an 8-bit unsigned integer.

(2) Contains the map following the specification as a string (**Warning:** the string is not \0-terminated!).

(3) Assigns a client a player number as an 8-bit unsigned integer.

(4) Is used to request a move from a player. It consists of an 32-bit unsigned integer indicating the timelimit (in $10^{-3}\,sec$), followed by an 8-bit unsigned integer indicating the maximum search depth (in both cases, 0 means no limit applies).

(5) Is used to respond with a move. It consists of the x and y coordinate as 16-bit unsigned integers, followed by an 8-bit unsigned integer for special fields. For choice fields, this number indicates the player number to switch with; for bonus fields either 20 for an extra bomb, or 21 for an extra override stone. For all other fields, this number must be 0.

(6) Is used to announce a made move to all players. It consists of the same fields as a move response (5), with an extra field for the number of the player that made the move, as an 8-bit unsigned integer.

(7) Announcement of the **disqualification** of a certain player, sent to all players. It consists of an 8-bit unsigned integer indicating the player that has been disqualified. After a player has been disqualified, no further messages will be sent to that client.

(8) Announces that the **first phase** has ended (and has no contents).

(9) Announces that the **second phase** has ended, and with that the **end of the game** (and has no contents). No further messages will be sent or received after this message.

Protocol

1. The client is provided the hostname (or IP-address) and port number of the server on the commandline.
2. Starting a connection with the server using a TCP connection.
3. The clients send their group number (1) to complete the connection.
4. When all clients are connected, the server will send them the map (2), followed by the player number (3).
5. In the **first phase** messages of type (4) through (7) will be sent: for requesting a move from a client (4) as well as the response to that (5); to inform clients of a made move (6); or the disqualification of a player (7).

- Message types (4), (6) and (7) are sent by the server, type (5) is sent by the client.
- After a move request (4), a player has to answer (5) within the given time limit. The client is only allowed to search up to and including the maximum depth specified. In case the time or depth limit is 0, then that limit does not apply.
- As soon as a player has made a move, the server will inform (6) **all** clients about this move.

6. As soon as no more player can make a move, the server will announce the end of the first phase (8). The second phase will start, with the same types of messages as in the first phase. In this phase, moves consist of throwing bombs.

7. If no more moves are possible in the second phase (all players used up all their bombs, of the map has no more tiles), then the server will announce the end of the second phase (9), and **the game ends**.

8. When an invalid move is sent, an invalid message is sent, or the connection is closed, the offending client will be **disqualified**. The server will inform all clients on the disqualification (7).

Example

Send bytes (hexadecimal)	Direction	Meaning
`01 00 00 00 01 17`	<<	Sending group number 23 to the server.
`02 00 00 00 34 32 0a 30 0a 31 20 31 0a 32 20 36 0a 2d 20 2d 20 62 20 78 20 32 20 2d 0a 2d 20 30 20 78 20 31 20 2d 20 2d 0a 31 20 31 20 35 20 3c 2d 3e 20 34 20 30 20 31 0a`	>>	Map of the server (52 bytes): 1 2 2 0 3 0 1 4 2 6 5 – – b x 2 – 6 – 0 x 1 – – 7 1 1 5 <–> 4 0 1
`03 00 00 00 01 02`	>>	Player number 2 assigned by the server.
...	...	Various messages.
`06 00 00 00 06 00 01 00 01 00 01`	>>	Player 1 made a move on tile $(1, 1)$.
`04 00 00 00 05 00 01 d4 c0 00`	>>	Move request with a $120000 \cdot 10^{-3}\,sec = 2\,min$ time limit and 0 depth limit – so no depth limit applies.
`05 00 00 00 05 00 02 00 00 14`	<<	Move response: place stone on bonus tile $(2, 0)$ while choosing a bomb.
`06 00 00 00 06 00 02 00 00 14 02`	>>	Move announcements: player 2 placed a stone on bonus tile $(2, 0)$ and got a bomb.
`08 00 00 00 00`	>>	End of the first phase reached – continuing with the second phase.
`04 00 00 00 05 00 01 d4 c0 00`	>>	Move request with a$120000 \cdot 10^{-3}\,sec = 2\,min$ time limit and 0 depth limit – so no depth limit applies.
`05 00 00 00 05 00 03 00 01 00`	<<	Move response: throw bomb at $(3, 1)$.
`06 00 00 00 06 00 03 00 01 00 02`	>>	Player 2 threw a bomb on tile $(3, 1)$.
`09 00 00 00 00`	>>	End of the second phase reached – end of the game.

Tips

- In Java the package `java.net` offers the basic network functionality. The package `java.io` offers classes/methods to easy processing of the transmitted data.
- In C/C++ you can include `sys/socket.h` to use network sockets. Please remember to convert between host and network byte order.