

General Game Playing

Prof. Michael Thielscher and Stephan Schiffel

International Masters Programme in Computational Logic — winter term 2006/07

30.10.2006

Exercise 2.1

Send me an email with your game description from exercise 1.2 (at least one from each team) by Sunday, 29 October, 23:59 CET. Make sure that your description abides by the GDL specification and that the game is playable (there is a legal move for each player in every reachable, non-terminal state) and (weakly) winnable for each player (for each player there is at least one sequence of joint moves such that the player wins).

Exercise 2.2

Start writing a program (in a programming language of your choice) that reads a game description in prefix-KIF format and is able to compute:

- a list of legal moves for all players given a particular state,
- the next state given a state and the move of each player,
- whether a given state is terminal, and what the value of the state is for each role in that case.

This task is for two exercises, that is, you have time until 06.11.2006.

Additional information:

- Specification of the GDL and example game descriptions:
<http://games.stanford.edu/language.html>
- Example programs in Lisp and Java: <http://games.stanford.edu/players.html>

Exercise 2.3

Implement the communication with the game manager. Once both tasks are done your program should be able to play any game with legal moves.

The infrastructure and communication protocol is described in this document:
http://games.stanford.edu/gdl_spec.pdf.

The protocol used for communication between the game manager and the players is a kind of http protocol. So basically each player is an http server that answers request from the game manager. There are three kinds of requests:

- START is sent when a new match is started and your player should reply with 'READY', when he is ready. With this request the game description is communicated to your player.
- PLAY is sent for every move. Your player has to send a reply with the move he wants to play. With this request the previous moves of all players are communicated to your player.
- STOP is sent when the match is over and your player should reply with 'DONE'.

We provide Java classes for the communication part of the player. So if you want to implement your player in Java you can use the classes provided on the course web page. However, it is easy to find free implementations of simple http-servers for a wide range of programming languages on the web. Often those can easily be adapted to the GGP communication protocol.

For testing your player you first have to log in to <http://games.stanford.edu:5101> and create a player for your team. After that you can create and start matches with your player. For multiplayer matches you can add 'Legal', 'Minimax' or 'Random' players.

As long as the reasoning part of the program isn't ready you can test the communication part by starting a match and sending 'NIL' (or any other string) as a reply to each PLAY message. It will be counted as an illegal move, but the game manager will pick a random move for you and the match will continue. At the end you should have received the same sequence of moves that is shown at the game manager web page for the match.