

Assignment 1a: Adversarial search, description and code submission page

- Inlämningsdatum 7 feb av 23.59
- Poäng 0,4
- Lämna in en filuppladdning
- Tillgänglig 25 jan kl 0:00–29 aug kl 23.59

Den här uppgiften låstes 29 aug kl 23.59.

Preliminary

This assignment is inspired by the TAI Game Playing Championship held for many years by late prof. Jan Eric Larsson from the EIT department at LTH.


In short:

Your task in this assignment consists in writing a program playing and winning a Connect Four game (aka 4-in-a-row). See below.


You may try your knowledge on something more challenging, namely Othello game, but the server offering it is unstable at the moment. The final announcement on its permanent availability will be posted on Friday, January 26th.

Closely after the deadline there will be a contest arranged for volunteers, so that you may test your solution against other students' solutions. Details come soon.


In detail:

The rules of Connect Four can be found, e.g., in https://en.wikipedia.org/wiki/Connect_Four  (https://en.wikipedia.org/wiki/Connect_Four), but you can find more resources on the web (including complete programs playing it).

In order to get the assignment passed, your program must:



1. Not loose against the course game server. As you know, Connect Four is a solved game and you may always at least get a draw. (<https://tromp.github.io/c4.html> , <https://tromp.github.io/c4.html>), 1990).
2. Not loose consistently, 20 games in a row. The server will store your id and the results of your games in order to verify this.
3. A smart method to achieve 1 and 2 is to use alpha-beta pruning in a minimax search for the best move. Your program is required to use it. The server uses this method to do it's moves (cutting search off after a depth of 3), so you might need to search deeper than that.

4. Your program should be fast, so a move needs to be decided within max 5 seconds. For that you might need to cut the search tree and introduce evaluation function of your choice. In order to ensure that, the server might behave unpredictably after waiting more than 5 seconds for your next move, e.g., might declare the game ended, with your loss.

You may write your program in any language you want. However, we provide a [Python template \(https://canvas.education.lu.se/courses/28438/files/4519586/download?wrap=1\)](https://canvas.education.lu.se/courses/28438/files/4519586/download?wrap=1)  (https://canvas.education.lu.se/courses/28438/files/4519586/download?download_frd=1) that would allow you to concentrate on the important stuff. In particular, the template contains the details of the protocol, server address, etc. You may find the short intro to the API [here \(https://canvas.education.lu.se/courses/28438/pages/api-assignment-1\)](https://canvas.education.lu.se/courses/28438/pages/api-assignment-1) as well.

Besides coding, the following demands must be fulfilled as well:

5. You must describe your solution in a report, **pointing to your choices related to items 3 and 4 above** and commenting your results;
6. You must include a user's guide instructing how to run your solution;
7. **IMPORTANT!** You must comment on some other student's solution (peer review). In order to get this step performed correctly you need to find a peer (student colleague, use the course discord chat for that) **well in advance before the deadline**, show her/him your solution, discuss your design choices, point to minimax/alpha-beta code in your solution and assess quality of your evaluation function. Then in **YOUR** report you need to take up your peer's solution and compare it to yours. You should consider asking [the following questions \(https://canvas.education.lu.se/courses/28438/pages/peer-review-assignment-1\)](https://canvas.education.lu.se/courses/28438/pages/peer-review-assignment-1) when doing the peer-review. Do not forget to put the name and STIL-ID of your peer in the report;
8. As the final part of the report you need to include a summary of your interaction with an AI chatbot of your choice (like Bing or ChatGPT) producing a solution to the assignment. You may need to ultimately fix it somehow, but try to force the AI system to get as close to the working solution as possible. If you have needed to fix the chatbot solution, summarize your additions/changes.

The submission, both your code, the chatbot code, and the report, needs to be done via canvas. Your code should be uploaded here, in Assignment 1a, while the report separately in Assignment 1b and the chatbot code in Assignment 1c. Please, typeset and format your report consistently, using Latex for instance. You may find the template file [here \(https://canvas.education.lu.se/courses/28438/files/4627563?wrap=1\)](https://canvas.education.lu.se/courses/28438/files/4627563?wrap=1)  (https://canvas.education.lu.se/courses/28438/files/4627563/download?download_frd=1) and its PDF version [here \(https://canvas.education.lu.se/courses/28438/files/4627562?wrap=1\)](https://canvas.education.lu.se/courses/28438/files/4627562?wrap=1). You may, for example, use [Overleaf](http://www.overleaf.com)  (http://www.overleaf.com) for that.

The deadline for this assignment is **Wednesday, 7th February 2024**, by which you **MUST** file in your working solution of the assignment, fulfilling **requirements 1-8**. As it involves peer review, plan well in advance!

You may, of course, consider further extensions of the game:

- A. another size of the playfield
- B. infinite playfield
- C. playing five-in-a-row
- D. implementing learning of the evaluation function
- E. reimplementing Alpha-Go approach
- F. Reading about and understanding the AlphaGo Zero

However, they are completely optional and should not make it into your report.

Important remark: Remember that the time frame allotted to this assignment is approximately five days of your work, so please don't overdo it: a decent program will suffice to get a pass. You are expected to write a simple program that uses a well-known adversarial search algorithm, provide it with some evaluation functions for guiding the search, and limit its search depth depending on time flow. Everything else would be an optional extra.

Please use only your own code. Lund University is committed to fighting every case of dishonesty or plagiarism.

We plan to use URKUND on your submissions. We may also use other plagiarism detection tools.

Please send any requests for clarification to Jacek.Malec@cs.lth.se or to the assignment 1 channel on discord server. The teaching assistants helping you with this assignment are Simon Kristoffersson Lind and Ayesha Jena, available on discord during the scheduled resource hours, together with two other TAs, Hashim and Eliot, who are available throughout the whole course. An email may also get reacted to, although without guarantees for short latency.