Planning and Optimization

A1. Organizational Matters

Gabriele Röger and Thomas Keller

Universität Basel

September 19, 2018

People & Coordinates

People: Lecturers

Gabriele Röger

Thomas Keller

Lecturers

Gabriele Röger

- email: `gabriele.roeger@unibas.ch`
- office: room 04.005, Spiegelgasse 1

Thomas Keller

- email: `tho.keller@unibas.ch`
- office: room 04.005, Spiegelgasse 1

People: Assistant

Guillem Francès

Assistant

Guillem Francès

- email: `guillem.frances@unibas.ch`
- office: room 04.004, Spiegelgasse 1

People: Tutors

Jendrik Seipp

Silvan Sievers

Tutors

Jendrik Seipp

- email: `jendrik.seipp@unibas.ch`
- office: room 04.001, Spiegelgasse 5

Silvan Sievers

- email: `silvan.sievers@unibas.ch`
- office: room 04.002, Spiegelgasse 1

Time & Place

Lectures

- time: Mon 14:15-16:00, Wed 14:15-16:00
- place: room 00.003, Spiegelgasse 1

Exercise Sessions

- time: Wed 16:15-18:00
- place: room 00.003, Spiegelgasse 1

first exercise session: today

Planning and Optimization Course on the Web

Course Homepage

`https://dmi.unibas.ch/de/studium/computer-science-informatik/lehrangebot-hs18/lecture-planning-and-optimization/`

- course information
- slides
- exercise sheets and materials
- bonus materials (not relevant for the exam)

registration:

- `https://services.unibas.ch/`
- Please register today to receive all course-related emails!

Target Audience & Rules

Target Audience

target audience:

- M.Sc. Computer Science/Informatik
  - "new" degree, Major in Machine Intelligence: module Concepts of Machine Intelligence
  - "new" degree, Major in Distributed Systems: module Applications of Distributed Systems
  - "old" degree: module Kerninformatik (core)
- M.A. Computer Science ("Master-Studienfach") module Concepts of Machine Intelligence
- other students welcome

Prerequisites

prerequisites:

- general computer science background: good knowledge of
  - algorithms and data structures
  - complexity theory
  - mathematical logic
  - programming
- background in Artificial Intelligence:
  - Foundations of Artificial Intelligence course (13548)
  - in particular chapters on state-space search

Gaps?

$\rightsquigarrow$ talk to us to discuss a self-study plan to catch up

Exam

- oral examination (20–25 min)
- dates: January 28–30
- 8 ECTS credits
- admission to exam: 50% of the exercise marks
- final grade based on exam exclusively
- no repeat exam

Exercise Sheets

exercise sheets (homework assignments):

- solved in groups of at most three $(3 < 4)$, submitted via Courses
- project-oriented assignments
  - each exercise sheet covers one part of the lecture
  - substantial in scope $\rightsquigarrow$ don't start too late
  - handed out at beginning of each part
  - work on these while we cover this part in the lecture
  - due six days after the end of the part
  - scope and marks proportional to covered topics
- mixture of theory, programming and experiments
- research aspects $\rightsquigarrow$ be independent, but ask questions!

Programming Exercises

programming exercises:

- part of regular assignments
- solutions that obviously do not work: 0 marks
- work with existing C++ and Python code
- Linux (other operating systems: vagrant virtual machine)
- pull from Mercurial (hg) repository

Exercise Sessions

exercise sessions:

- discuss past homework assignments
- ask questions about current assignments (and course)
- work on homework assignments
- sometimes live exercises

Plagiarism

Plagiarism (Wikipedia)

> *Plagiarism is the "wrongful appropriation" and "stealing and publication" of another author's "language, thoughts, ideas, or expressions" and the representation of them as one's own original work.*

consequences:

- 0 marks for the exercise sheet (first time)
- exclusion from exam (second time)

if in doubt: check with us what is (and isn't) OK before submitting

exercises too difficult? we are happy to help!

Course Content

Learning Objectives

Learning Objectives

- get to know theoretical and algorithmic foundations
  of classical & probabilistic planning
  as well as practical implementation
- understand fundamental concepts underlying modern planning algorithms and theoretical relationships that connect them
- become equipped to understand research papers
  and conduct projects in this area

Course Material

course material:

- slides (online + printed handouts)
- no textbook
- additional material on request

Hands-On Week

- Next week will be a hands-on week.
- Please bring your laptop to next week's sessions (Monday and Wednesday).

Don't own a laptop?

- no problem, we will do the hands-on in groups of 3

Today’s Exercise Session

- To make the hands-on week work smoothly, we try to work out compilation issues etc. today in the exercise session.
- The goal of today’s exercise session is that you can run the examples of today’s lecture on your own machine.
- The following slide contains the main information for today’s setup for your future reference.
- In any case, please complete the setup before next Monday.
- We are happy to help you if you run into problems.

Your First Tasks (1) – on Ubuntu

Getting Started: Cloning the Repository

Install mercurial (if not already installed):

```
sudo apt install mercurial
```

Clone the course repository:

```
hg clone https://bitbucket.org/aibasel/planopt-hs18
```

Enter demo directory:

```
cd planopt-hs18/classical/demo
```

Your First Tasks (1) – using vagrant

Assumption: virtual box, vagrant, X server and SSH client available
console in new directory, containing file `Vagrantfile`

Getting Started: Setting up virtual machine

Set up virtual machine:

```
vagrant up
```

Login:

```
vagrant ssh
```

Enter demo directory:

```
cd planopt-hs18/classical/demo
```

Your First Tasks (2)

Getting Started: Building Fast Downward

Build Fast Downward and set a symbolic link:

```
cd fast-downward
./build.py
cd ..
ln -s fast-downward/fast-downward.py .
```

Without virtual machine

- See build instructions and dependencies at: `http://www.fast-downward.org/ObtainingAndRunningFastDownward`.
- Note that we use our own repository, not `hg.fast-downward.org`.
- You can skip the optional information regarding the LP solver.

Test `fast-downward.py` with the examples in the next chapter.

Your First Tasks (3)

Getting Started: Building VAL

Build VAL and set a symbolic link:

```
cd VAL
make -j4
sudo ln -sf `pwd`/validate /usr/bin/validate
```

Without virtual machine

- The main dependencies of VAL are `g++`, `make`, `flex` and `bison` (Ubuntu package names).

Test `validate` with the examples in the next chapter.

Under Construction. . .

- This year we will newly cover probabilistic planning.
- We are always happy about feedback, corrections and suggestions!