Grundlagen der Künstlichen Intelligenz

Programming Exercise 2: Constraint Satisfaction Problem 22nd November 2019

Anna-Katharina Rettinger, Lennart Aigner, (last updated 22nd November 2019)
Jiaying Huang and Meric Sakarya

Problem 2: Science Fair

For the science fair at Python Elementary, Ms. Lovelace assigns her students different projects to work on and present at the end of the semester. The students she is teaching are: famous girl of the school Bella, captain of the volleyball team Bethany, funny twins of the class Brian and Brianna, the quiet one Ben, the science guy Bill and the mischievous one Bart. Ms. Lovelace wants this to be the best science fair Python Elementary has seen. For this she wants everyone to work on a project and learn something new about the mysterious world of science:

- Solar Powered Car
- Potato Battery
- Baking Powder Volcano
- Our Solar System

Consider the following constraints:

1. Everyone has a project
2. No one presents alone
3. Every project is presented by someone (i.e. each project must be presented at least by one student)
4. Up to 4 students may work on the Solar Powered Car
5. Up to 3 students may work on the Potato Battery
6. At least 3 and at most 5 students may work on the Baking Powder Volcano
7. Our Solar System is a project for 2 students
8. Bart presents together with Bethany
9. Brian, Bill and Ben don't want to present together because they don't get along
10. Bill and Brianna don't want to present the Baking Powder Volcano because it's boring
11. Bella and Bill present the Potato Battery
12. Bella presents with at least another girl beside herself
13. No one presents Our Solar System
14. Brian and Brianna present together

Model the constraint satisfaction problem in Python. For each of the following subsets of constraints, find the solution, if it exists:

Problem 2.1: { 1 – 2, 4 – 12 }

Problem 2.2: { 1, 3 – 10, 12 }

Problem 2.3: { 1, 3 – 7, 10 – 12 }

Problem 2.4: { 1 – 2, 4 – 7, 10 – 13}

Problem 2.5: { 1 – 2, 4 – 11, 13 }

Problem 2.6: { 1 – 2, 4 – 7, 10 – 14}

Note that problems `2.1` and `2.3` can not be satisfied.

Programming Framework

For this programming exercise *Jupyter Notebooks* will be used. The template for the exercise can be found in Moodle. Since you only have to model the constraint satisfaction problem, only minor programming skills in Python are necessary to complete this exercise. The following steps are required to correctly set up the environment for the programming exercise:

1. **Installation of Anaconda:** If you do not already have the *Jupyter Notebook* environment installed on your machine, the installation is the first step you have to perform. We recommend to install *Anaconda*, since this will set up the whole environment for you. Instructions on how to install *Anaconda* can be found in the "*AIMAcode Installation Instructions*" file in Moodle.
2. **Download of the AIMA python code:** The template for the programming exercise is based on the code from the *AIMAcode* project. Therefore, you first have to download the code from this project before the template can be used. Instructions on how to obtain the code from this project can be found in the "*AIMAcode Installation Instructions*" file in Moodle.
3. **Download of the template:** Download the .zip-folder with the template from Moodle. To avoid issues with the relative file paths, we recommend to copy all files contained in the .zip-folder into the root-directory of the *AIMAcode* project that you downloaded in the previous step.

After completing the above steps, you are all set up to start with the exercise. The main function of the template is the *Jupyter Notebook* **AI_Assignment2.ipynb**, which is also the only file you have to work on. Your task is to model the Science Fair problem. An example, on how to model a constraint satisfaction problem using the *AIMAcode*, is provided in the notebook.

Submission

For submission, you have to upload the following files in Moodle:

1. **AI_Assignment2.ipynb** – notebook containing your solution for modelling the Science Fair problem.
2. **AI_Assignment2.py** – python script containing the same implementation. (Download the AI_Assignment2.ipynb as a python script within your juypter notebook environment: File → Download as → Python (.py))

A pass will be awarded only if:

1. you submitted the correct files with the correct names, as shown above.
2. you did not change the variable names provided by us.
3. your submitted files can be run in an Anaconda environment (Python 3.6 or 3.7) with the packages provided by the *requirements.txt* in the *aima repository* and the *csp_programming_exercise.py* provided by us.
4. the problem has been modelled correctly.
5. like the rest of the programming exercises, this is an individual project and work **must** be your own. (We will use a plagiarism detection tool and any copied code will annul all bonus exercises from both the copier and the copied person!)

Submission will close on **Sunday, 15.12.2019 at 23:59**. Your solution will be marked using a shell script. You can either pass (if all requirements listed above are met) or fail (unfortunately, we cannot manually check for minor errors). Thus, it is very important to follow the instructions exactly!

We offer a preliminary check of your solution: All solutions uploaded until Monday, 09.12.2019 at 23:59 will be checked for formatting errors, i.e., whether your code runs on our machine and you submitted the correct files. An automated feedback will be submitted via Moodle until Wednesday, 11.12.2019 so that you can (re)submit a formally correct solution until the main submission deadline.