Computer Science 2542 June 1, 2014
St. George Campus University of Toronto

Assignment #1
Warm-up Assignment
**Part I Due: Monday, June 30, 2014, noon** (at the tutorial)
**Part II Due: Monday, July 7, 2014, noon** (at the tutorial)

---

**Late Penalty:** Late assignments will be penalized 10% for each day they are late.

**Total Marks**: There are 100 marks available in this assignment. This assignment represents 20% of the course grade.

**Advice:** Part II is worth 45 marks. If you postpone starting until June 30, you'll be in trouble. It will take you some time to devise an approach, test it, and do the experimental analysis. This is the fun (and challenging) part of the assignment. Give yourself time to enjoy it!

**Handing in this Assignment:** Each question already describes what to hand in. Everything has to be submitted in a written report unless the question explicitly requests that it be submitted by e-mail. Please make sure that source files for individual questions are clearly distinguishable and attributable to you. Please provide precise instructions for compilation.

Material submitted by e-mail should be sent to edelisle/AT/cs.toronto.edu

**Further Material and Information** Additional material and information will be posted on the course's web page (follow the link to Assignment 1). This includes:

1. The FF planner source code (in C, with special thanks to Jorge Baier). **You must use these sources; *not* the ones available from the author.** In the past we have found the official FF source code to have significant bugs, in addition to compatibility issues with up-to-date operating systems. These problems do not exist in our sources. Moreover, our sources contain some useful comments that you will likely appreciate.
2. A brief FF guide, explaining relevant data structures and functions.
3. The domains and problem files that you'll use to evaluate the planner.
4. A brief tutorial on PDDL, the specification language for the domains and problem files.
5. Important clarifications and hints may be posted after the assignment has been handed out. So keep watching the web page! Notification of the existence of new information will also be logged on the course announcements page.

## References

[1] Blai Bonet and Hector Geffner. Planning as heuristic search. *Artificial Intelligence*, 129(1-2):5–33, 2001.

[2] Jörg Hoffmann and Bernhard Nebel. The FF planning system: Fast plan generation through heuristic search. *Journal of Artificial Intelligence Research*, 14:253–302, 2001.

## Warm-up Assignment

This assignment is a warm-up to get your hands dirty with an implemented planner and some International Planning Competition (IPC) test domains. The planner you will be working with is Fast-Forward (FF), a heuristic search planner by Hoffmann and Nebel [2].

**PART I**

**Question 1 [25 pts]:** The STORAGE domain was introduced for the 2006 International Planning Competition. It consists of a number of hoists that can be used to move crates, which are initially located in containers, to depots. The depots are divided into different locations (called *store-areas*). Hoists can move through store-areas as long as they are clear, and can drop a crate in a clear store-area. Store-areas are not clear if either a hoist or a crate is on it. For more details, take a look at the PDDL specification of the domain.

STORAGE is a very simple domain. Indeed it can be solved (non-optimally) in polynomial time. However, FF cannot solve many of the instances.

1. Give a precise explanation for this behaviour. Justify your answer by performing a small experiment. (*Hint:* Focus on problem 19 to answer the question. You might need to modify FF to show more information.)

2. Write a new version of the STORAGE domain that is easier for FF to solve on the harder problems (it doesn't matter if some of the easy problems are now unsolvable). Your version of the STORAGE domain must satisfy the following conditions: (1) Plans in your version should also be plans for the original version. (2) It should differ from the original STORAGE domain at most in the definition of one action.

***What to hand in for Question 1:*** *A description of the experiment that led you to your conclusions. Your modified* STORAGE *domain file (both on paper and by e-mail). A table showing which problems are solved in your version of the* STORAGE *domain.*

**Question 2 [30 pts]:** When building a relaxed plan, FF uses a heuristic rule to choose the achiever of a subgoal when there exist multiple achievers. It chooses the achiever with the least cost, where the cost of an achiever *a* is given by the following expression.

$$C(a) = \sum_{f \in \mathsf{Prec}(a)} \mathsf{level}(f),$$

where $\mathsf{level}(f)$ is the level in the relaxed graph at which $f$ first appears, and $\mathsf{Prec}(a)$ is the precondition fact list for *a*.

This heuristic rule favours actions that have preconditions that are "easy to achieve". However, estimating the precondition cost by the level of its facts is a myopic decision. Consider the following cost function for an action *a* in a relaxed graph constructed from *s*:

$$C(a) = 1 + \sum_{f \in \mathsf{Prec}(a)} C_F(f), \text{ such that}$$

$$C_F(f) = \begin{cases} 0 & \text{if } f \text{ holds true in } s \\ \min_{a \in \mathsf{FirstAchievers}(f)} C(a) & \text{otherwise,} \end{cases}$$

where $\mathsf{FirstAchievers}(f) = \{a \,|\, f \in \mathsf{Adds}(a) \text{ and } \mathsf{level}(a) = \mathsf{level}(f) - 1\}$, is intuitively the set of actions that add $f$ for the first time in the plan graph. $\mathsf{level}(a)$ is the level at which action *a* first appears in the graph, and $\mathsf{Adds}(a)$ is the add-list of *a*.

Intuitively this function estimates the cost of a precondition by choosing the cost of the best action that can satisfy it. (This cost function is based on the $h_{add}$ heuristic by Bonet and Geffner [1].) $C(a)$ can be computed rather straightforwardly while building the relaxed graph.

Implement this new heuristic rule, run the planner on all the planning domain and problem provided on the assignment web page, and conclude whether or not this is a better rule in terms of search effort and solution quality.

***What to hand in for Question 2:*** *Besides your analysis, we need tables reporting, for each problem in each domain, number of nodes evaluated and number of actions in the solutions you found with the new action selection strategy. Graphs are not required. We require you to send, by e-mail only, the source files that you modified.*

**PART II**

**Question 3 [45 pts]:** Implement a new heuristic rule for action selection in the relaxed graph construction that considers in some way the deletes of the actions in the relaxed plan. Run the planner on all our planning tasks, and evaluate the technique.

Your technique should be intuitively sensible, easy to describe, but is not required to beat the FF heuristic. The mark will depend on: (1) the merit of the approach, and (2) experimental results in relation to those obtained by other students in the class.

***What to hand in for Question 3:*** *Same as in Question 2 plus a detailed description of your approach.*