UNIVERSITÄT HEIDELBERG
Zukunft. Seit 1386.
IWR Interdisciplinary Center for Scientific Computing

Artificial Intelligence
Wintersemester 17/18
Prof. Björn Ommer
ommer@uni-heidelberg.de

Exercise sheet 3: Informed search algorithms

Due on *17/11/2017*, 2pm.

Question 1: Simulated Annealing (3P)

a) Explain the difference between the Simulated Annealing Search and the Local Hill-Climbing Search in your own words. Give one simple example case in which the two search algorithms show different outputs.

b) Consider the function

$$f(x) = 0.2\sin(12.5x) + (x-1)^2 - 5.$$

We discretize these data evaluating $f(x)$ at 31 points, resulting in a vector $f = \{x_1, x_2, \ldots, x_{31}\} \in \mathbb{R}^{31}$.

The task is to find the global minimum by simulated annealing.

Implement a simulated annealing approach consisting of a loop with not more than 300 iterations. A temperature T must be successively decreased (e.g. by $T \leftarrow \alpha T, \alpha < 1$; here you need to tune α). Start with index $current\_pos = 0$ and each iteration algorithm should move to the left or to the right along axis X. Index $current\_pos$ is replaced by a random index j (where $j = current\_pos - 1$ or $j = current\_pos + 1$), if either $f_j < f_i$ or with a probability p given by

$$p(current\_pos \to next) = \exp\left(-\frac{f_{next} - f_{current\_pos}}{T}\right) \qquad (\text{if } f_{next} \geq f_{current\_pos}).$$

Keep track of the best index at every iteration.

File `ex1.b.py` contains a starter code. You need to implement the function `minimize()`.

Question 2: A* Search Algorithm (5.5P)

Implement A* search algorithm for pathfinding in a graph.

Try different heuristics. Which one is better?

Prof. Björn Ommer
ommer@uni-heidelberg.de

File `ex2_search.py` contains a starter code. You need to implement a class `PriorityQueue` and functions `heuristic()` and `a_star_search()`.

Your solution will be evaluated on a series of tests. A subset of tests are available in folder `grids`. After submission the solution will be evaluated on the full test set.

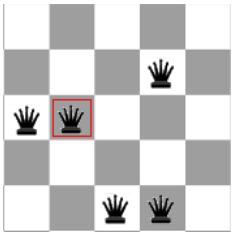
Your implementation must be effective. Tests check the optimality of the found path and the speed of your implementation.

It is strongly recommended to run local tests using `ex2_utest.py` before submission. All tests should be passed. Use script `ex2_run_one_test.py` for visualizing the result of your solution and debugging.

Prof. Björn Ommer
ommer@uni-heidelberg.de

Question 3: Local Search **(1.5P)**

We consider the 5-queens problem with the following configuration:



We say that the cost $f(n)$ of a configuration n is the number of conflicts between the queens, counting the case that one queen attacks another while the other one also attacks the former, as one conflict. Thus, in the above configuration there are 5 conflicts. Now assume that the marked queen can freely move to any free field on the board. This would change the number of conflicts.

a) Provide for any new position of the marked queen the cost value f of the new configuration. (Find any easy way to calculate f only from the conflicts of that queen plus a constant number.)
b) Describe the move of the queen resulting from local search. What would be the next steps for local search (in words)?

Note: Submit exactly one ZIP file and one PDF file via Moodle before the deadline. The ZIP file should contain scripts `ex1.b.py` *and* `ex2_search.py`*, it* ***shouldn't contain any folders****. Make sure that it runs on different operating systems and use relative paths. Non-trivial sections of your code should be explained with short comments, and variables should have self-explanatory names. The PDF file should contain your written code, all figures, explanations and answers to questions. Make sure that plots have informative axis labels, legends and captions.*