Introduction to Artificial Intelligence: Assignment 4 - Simulated Annealing

Andreas Dørum, Martin Baklid

October 2015

1 Introduction

2 Implemenation

To specialize the simulated annealing algorithm for use with the egg carton problem, we need three things: a representation of solutions, and objective function, and a method for generating neighbors.

We choose to represent solutions as a set of tuples with x and y coordinates for each egg.

The objective function needed to rate to aspects of the solution, both the number of eggs, more being better, and "overflow": the number of eggs beyond the k allowed in each row, column and diagonal.

For eggs, the number is counted and divided by the total number possible for the carton in question. For the boards in this assignment it hold that the maximum possible, and hence the optimal, is min(n, m) * k.

For collisions we returned the the maximum number of collisions in any row, column or diagonal. The overflow is the number of collision more than k, F returns overflow negated.

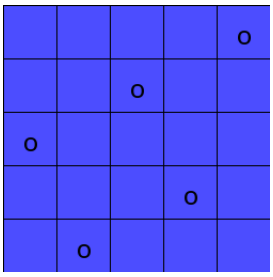
$$F = \begin{cases} \frac{egg}{egg_{tot}}, & \text{if } collisions < k \\ -(collisions - k), & \text{otherwise} \end{cases} \tag{1}$$

Neighbors are considered to be cartons with one more or one less egg than the current one. The neighbor-function flips a coin whether to add to or remove from the current state. When adding, a random position tuple is picked from the set of free positions. When removing, a random tuple is removed from the set of current eggs.

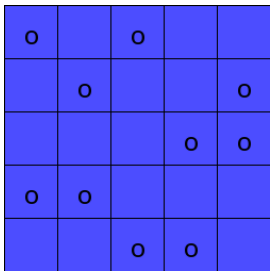
3 Diagrams

4 Question

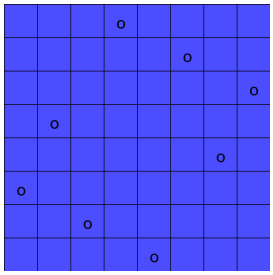
Heuristics and objective functions are similar in that they rate the "goodness" of a given state. Both of them require specialized knowledge to pass judgement. The difference is that while an objective function rates a known state, the heuristic makes a guess on an unknown state using domain knowledge. The states that they rate are also different: The heuristic estimates how promising a *partial* solution is, while the objective function rates the quality of a *complete* solution.



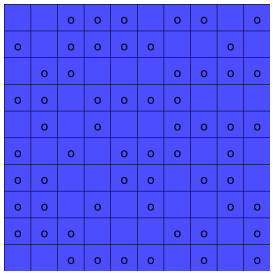
(a) SA, m = 5, n = 5, k = 2



(b) SA, m = 6, n = 6, k = 2



(c) SA, m = 8, n = 8, k = 1



(d) SA, m = 10, n = 10, k = 6