CS4618 Artificial Intelligence I
Term 2012/2013, 1st Period
http://www.cs.ucc.ie/∼tj2/cs4618/

Thomas Jansen
University College Cork
October 17th, 2012

Assignment 4

Topic 'Local Search'

Problem 1 We want to approach MAX-SAT by means of local search. In MAX-SAT we are given a set of clauses over variables and are looking for an assignment of the variables that maximises the number of clauses that are true. We define this in the following more precisely.

Variables are x_1, x_2, ..., x_n. Each variable may assume one of the values 1 or 0 (corresponding to true and false). A literal is a variable or a negated variable. For example, we write the negation of x_7 as $\overline{x_7}$. If a variable is 0, its negation is 1. If a variable is 1, its negation is 0.

A clause is a disjunction of literals. For example, we write the disjunction of x_1, $\overline{x_3}$ and x_6 as $x_1 \vee \overline{x_3} \vee x_6$. A clause has the value 1 if at least one of its literals has the value 1, otherwise the clause has value 0.

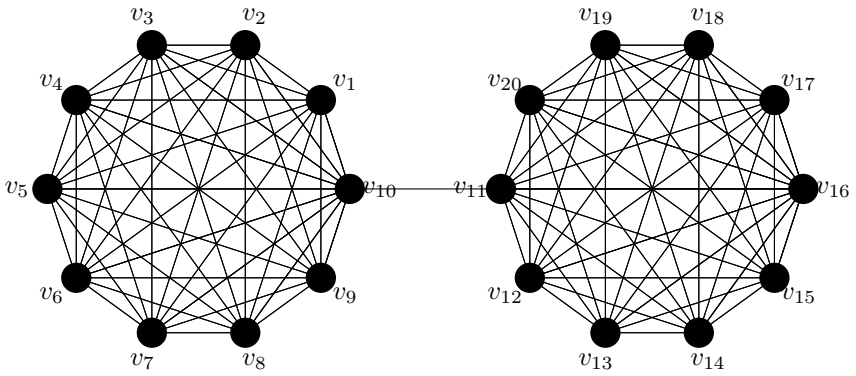
In MAX-SAT we want to set the variables to values in such a way, that the number of clauses that have value 1 is maximised.

We consider the following nine clauses over the variables x_1, x_2, x_3.

$c_1 \colon \overline{x_1}$, $c_2 \colon \overline{x_2}$, $c_3 \colon x_3$, $c_4 \colon \overline{x_1} \vee x_2$, $c_5 \colon x_1 \vee \overline{x_2}$, $c_6 \colon x_1 \vee x_3$, $c_7 \colon \overline{x_2} \vee \overline{x_3}$, $c_8 \colon x_1 \vee x_2 \vee x_3$, $c_9 \colon x_1 \vee \overline{x_2} \vee x_3$

We want to use first improvement local search to find an assignment for this instance of MAX-SAT. Use $x_1 = 0$, $x_2 = 0$, $x_3 = 0$ as the initial assignment. Two assignments are neighbours if they differ in the assignment of exactly one variable. When going through the neighbourhood change x_1 before x_2 and x_2 before x_3. Write down all assignments you consider and for each the assignment the number of clauses that have value 1. Stop when no better assignment is found.

Topic 'Evolutionary Algorithms'



Problem 2 For $k \in \mathbb{N}$ we define the following undirected graph $G = (V, E)$ by its set of nodes $V = \{v_1, v_2, \ldots, v_{2k}\}$ and its set of edges $E = \{\{v_i, v_j\}, \{v_{k+i}, v_{k+j}\} \mid 1 \leq i < j \leq k\} \cup \{\{v_k, v_{k+1}\}\}$. An example for $k = 10$ is shown above.

For $k \in \mathbb{N}$ we define $n := 2k$ and a function $f\colon \{0,1\}^n \to \mathbb{R}$ in the following way where $x = x[1]x[2]\cdots x[n] \in \{0,1\}^n$.

$$f(x) := x[k]\cdot x[k+1] + (1-x[k])\cdot(1-x[k+1]) + \left(\sum_{i=1}^{k-1}\sum_{j=i+1}^{k} x[i]\cdot x[j] + (1-x[i])\cdot(1-x[j])\right) + \left(\sum_{i=1}^{k-1}\sum_{j=i+1}^{k} x[k+i]\cdot x[k+j] + (1-x[k+i])\cdot(1-x[k+j])\right)$$

The function can be explained in the following way. In the graph G each node can be 'coloured' using the 'colours' 0 or 1. The function f yields the number of edges such that the two nodes of an edge are coloured using the same colour.

Discuss what you expect the (1+1) EA to do on this function (if k is not very small, say $k \geq 10$). Discuss why algorithms that use 1-point crossover may be more successful.

Problem 3 **Implementation Task**

Implement random local search, the Metropolis algorithm and simulated annealing within a single framework. The search space is $\{0,1\}^n$. Two bit stings are neighbours if they differ in exactly one bit.

Implement the framework in a way that you can re-use modules that appear in different algorithms. Moreover, implement it in a way that you can set the number of times each algorithm is run as a parameter.

The function f needs to be implemented as a module. This module needs to return the function value and keep track of the number of times it has been called.

We use this framework with functions f where we know an optimal value. Let each algorithm run until a solution with this optimal value is found. The framework needs to keep track of the number of times each algorithm needed to compute the f-value in order to locate an optimal value. It returns as output the average of these numbers (where the average is over the different runs).

Consider the simple fitness function $f(x) := \sum_{i=1}^{n} x[i]$. For random local search and the Metropolis algorithm do 30 runs for $n \in \{10, 20, 30, 40, 50, 60, 70, 80, 90, 100\}$. Feel free to experiment with the values for the temperature. Draw a graph where you have n on the x-axis and the average number of function values on the y-axis.

Assignments are handed out on each Friday during the lecture. Written solutions are due the next Wednesday. Feedback is given and solutions are discussed the Wednesday after that.