Heuristic Optimisation

Part 9: Tabu search

Sándor Zoltán Németh

http://web.mat.bham.ac.uk/S.Z.Nemeth

s.nemeth@bham.ac.uk

University of Birmingham

Overview

- The main idea of tabu search
- Tabu search for SAT
- Tabu search for TSP

Introduction

Proposed independently by Glover (1986) and Hansen (1986)

> ”a meta-heuristic superimposed on another heuristic. The overall approach is to avoid entrapment in cycles by forbidding or penalizing moves which take the solution, in the next iteration, to points in the solution space previously visited (hence tabu).”

Basic features

- Accepts non-improving solutions deterministically in order to escape from local optima (where all the neighbouring solutions are non-improving) by guiding a hill-climbing algorithm

- Uses memory in two ways:

  to prevent the search from revisiting previously visited solutions

  to explore the unvisited areas of the solution space

Basic features cont'd

- Uses past experiences to improve current decision making

- The use of a memory (a ”tabu list”)

  – to prohibit certain moves –

  makes tabu search a global optimiser rather than a local optimiser

Tabu search for SAT

Consider a SAT problem of 8 variables

The initial assignment is $\mathbf{x} = (0, 1, 1, 1, 0, 0, 0, 1)$

The neighbourhood of $\mathbf{x}$ (obtained by flipping the variables' values, one at a time) is examined

The best neighbour, obtained by flipping the third variable, is selected

Building up the memory

When flipping a variable we ”make a note” of it, so that the same variable is not flipped a predefined number of steps (for ex. 5)

0	0	5	0	0	0	0	0

The *i*th variable is tabu for *j* steps/iterations/moves

In the next step:

- some variable (other than the third) is selected to be flipped
- the memory is updated, i.e., all non-zero values are decremented

Using the memory

After 5 iterations the memory is:

3	0	1	5	0	4	2	0

$\mathbf{x} = ?$

The "available" variables are x_2,x_5 and x_8

If x_5 is the best, after the 6th iteration the memory becomes:

2	0	0	4	5	3	1	0

Example

$$F = (x_1 \bigvee x_2 \bigvee x_3) \bigwedge (\overline{x}_2 \bigvee \overline{x}_3 \bigvee x_4) \bigwedge x_2 \bigwedge x_3 \bigwedge (\overline{x}_4 \bigvee x_5)$$

Memory	x_1	x_2	x_3	x_4	x_5	F_1	F_2	F_3	F_4	F_5	#	decrease
00000	0	0	0	0	0	0	1	0	0	1	3	–
	1	0	0	0	0	1	1	0	0	1	2	1
	0	1	0	0	0	1	1	1	0	1	1	2←
	0	0	1	0	0	1	1	0	1	1	1	2
	0	0	0	1	0	0	1	0	0	0	4	-1
	0	0	0	0	1	0	1	0	0	1	3	0
03000	0	1	0	0	0	1	1	1	0	1	1	–
	1	1	0	0	0	1	1	1	0	1	1	0←
	0	1	1	0	0	1	0	1	1	1	1	0
	0	1	0	1	0	1	1	1	0	0	2	-1
	0	1	0	0	1	1	1	1	0	1	1	0
32000	1	1	0	0	0	1	1	1	0	1	1	–
	1	1	1	0	0	1	0	1	1	1	1	0←
	1	1	0	1	0	1	1	1	0	0	2	-1
	1	1	0	0	1	1	1	1	0	1	1	0
21300	1	1	1	0	0	1	0	1	1	1	1	–
	1	1	1	1	0	1	1	1	1	0	1	0←
	1	1	1	0	1	1	0	1	1	1	1	0
10230	1	1	1	1	0	1	1	1	1	0	1	–
	1	0	1	1	0	1	1	0	1	0	2	-1
	1	1	1	1	1	1	1	1	1	1	0	1←

Extensions

Aspiration criterion:

In specific circumstances an ”outstanding” tabu can be accepted as the next point to be visited

The memory discussed so far is recency-based

Frequency-based memory can be used to diversify the search:

$$H(i) = j$$

”during the last h iterations of the algorithm variable i was flipped j times”

Tabu search for TSP

Neighbourhood can be defined as swapping two cities in a tour

For a TSP problem with 8 cities, a tour
$(2, 4, 7, 5, 1, 8, 3, 6)$ will have 28 neighbours

The structure of the recency-based memory:

2	3	4	5	6	7	8	
							1
							2
							3
							4
							5
							6
							7

TSP example

The current tour after 500 iterations is
$(7, 3, 5, 6, 1, 2, 4, 8)$

The recency-based memory

2	3	4	5	6	7	8	
0	0	1	0	0	0	0	1
	0	0	0	5	0	0	2
		0	0	0	4	0	3
			3	0	0	0	4
				0	0	2	5
					0	0	6
						0	7

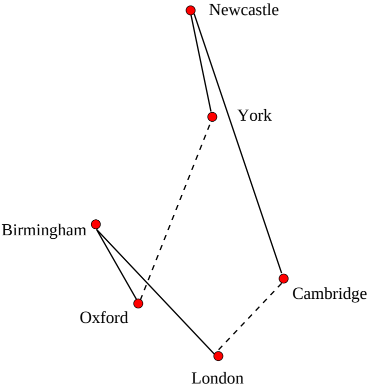
TSP example cont'd

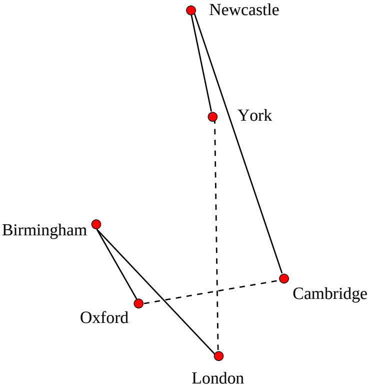
The frequency-based memory

2	3	4	5	6	7	8	
0	2	3	3	0	1	1	1
	2	1	3	1	1	0	2
		2	3	3	4	0	3
			1	1	2	1	4
				4	2	1	5
					3	1	6
						6	7

This is not the best neighbourhood definition

Neighbourhood based on 2-interchange





Tabu search based on 2-interchange

Repeat MAX-TRIES times:

1. Generate a tour
2. Repeat ITER times
   (a) identify a set of 2-interchange moves
   (b) select & make a 2-interchange
   (c) update tabu list
   (d) update local best tour info
3. Update global best tour info

Pros and cons for tabu search

Pros:

Generally good solutions for optimisation problems

Cons:

Tabu list construction is problem specific

No guarantee of global optimum

Tabu search and simulated annealing

- Both were designed to escape local optima
- Both work on complete solutions

- Tabu search only selects worse moves if it is stuck, whereas simulated annealing can do that all the time

- Simulated annealing is stochastic
- Tabu search is deterministic

- The parameters must be carefully chosen for both

Recommended reading

Z. Michalewicz & D.B. Fogel
How to Solve It: Modern Heuristics

Section 5.2 Tabu search