G5BAIM
Artificial Intelligence Methods

Dr. Rong Qu

Tabu Search

Characteristics of SA (review)

- Random selection of a neighbouring solution
- Probabilistic acceptance of non-improving solutions;
- The best solution is recorded:
  - Lack of memory of history of search;
  - All the information found during the search is lost;

Tabu Search

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*)."

Tabu Search (continued)

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

Tabu Search (continued)

- After evaluating a number of neighbourhoods, we accept the best one, even if it is low quality on cost function.
  - Accept worse move
- Uses of memory in two ways:
  - prevent the search from revisiting previously visited solutions;
  - explore the unvisited areas of the solution space;
  - for example,

Tabu Search (continued)

- Use past experiences to improve current decision making.
- By using memory (a "tabu list") to prohibit certain moves - makes tabu search a **global** optimizer rather than a local optimizer.

Tabu Search vs. Simulated Annealing

- Accept worse move
- Selection of neighbourhoods
- Use of memory

Is memory useful during the search?

Uses of memory during the search?

- Intelligence needs memory!
- Information on characteristics of good solutions (or bad solutions!)

Uses of memory during the search?

- Tabu move – what does it mean?
  - Not allowed to re-visit exact the same state that we've been before
    - Discouraging some patterns in solution: e.g. in TSP problem, tabu a state that has the towns listed in the same order that we've seen before.
    - If the size of problem is large, lot of time just checking if we've been to certain state before.

Uses of memory during the search?

- Tabu move – what does it mean?
  - Not allowed to return to the state that the search has just come from.
    - just one solution remembered
    - smaller data structure in tabu list

Uses of memory during the search?

- Tabu move – what does it mean?
  - Tabu a small part of the state
    - In TSP problem, tabu the two towns just been considered in the last move – search is forced to consider other towns.

Uses of memory during the search?

- In Tabu Search
  - What neighbourhood is
  - What constitute a tabu list

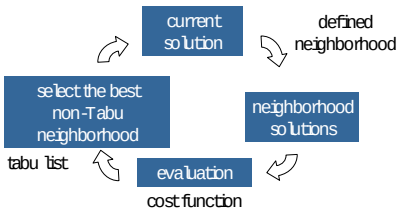
Dangers of memory

- Exhaustive usage of memory resources
  - Design of efficient data structures to record and access the recorded data efficiently;
    - Hash table
    - Binary tree
- Memorising information which should not be remembered

Dangers of memory

- Collecting more data than could be handled:
  - Clear understanding of which attributes of solutions are crucial;
  - Limited selection of attributes of solutions to be memorised;
  - Clear strategy on usage of information or their disposal when not needed;

Tabu Search



Tabu Search algorithm

Function TABU_SEARCH(*Problem*) **returns** a solution state

- **Inputs**: *Problem*, a problem
- **Local Variables**:
  - *Current*, a state
  - *Next*, a state
  - *BestSolutionSeen*, a state
  - *H*, a history of visited states

Tabu Search algorithm (cont.)

- *Current* = MAKE-NODE(INITIAL-STATE[*Problem*])
- **While not terminate**
  - *Next* = a highest-valued successor of *Current*
  - **If(not** Move_Tabu(*H,Next*) **or** Aspiration(*Next*)**) then**
    - *Current* = *Next*
    - Update *BestSolutionSeen*
    - *H* = Recency(*H* + *Current)*
  - Endif
- **End-While**
- **Return** *BestSolutionSeen*

Elements of Tabu Search

- Memory related - recency (How recent the solution has been reached)
  - Tabu List (short term memory): to record a limited number of attributes of solutions (moves, selections, assignments, etc) to be discouraged in order to prevent revisiting a visited solution;
  - Tabu tenure (length of tabu list): number of iterations a tabu move is considered to remain tabu;

Elements of Tabu Search

- Memory related – recency (How recent the solution has been reached)
  - Tabu tenure
    - List of moves does not grow forever – restrict the search too much
    - Restrict the size of list
    - FIFO
    - Other ways: dynamic

Elements of Tabu Search

- Memory related – frequency
  - Long term memory: to record attributes of elite solutions to be used in:
    - Diversification: Discouraging attributes of elite solutions in selection functions in order to diversify the search to other areas of solution space;
    - Intensification: giving priority to attributes of a set of elite solutions (usually in weighted probability manner)

Elements of Tabu Search

- If a move is good, but it's tabu-ed, do we still reject it?
- Aspiration criteria: accepting an improving solution even if generated by a tabu move
  - Similar to SA in always accepting improving solutions, but accepting non-improving ones when there is no improving solution in the neighbourhood;

Example: TSP using Tabu Search

In our example of TSP:

Find the list of towns to be visited so that the travelling salesman will have the shortest route

- Short term memory:
  - Maintain a list of t towns and prevent them from being selected for consideration of moves for a number of iterations;
  - After a number of iterations, release those towns by FIFO

Example: TSP using Tabu Search

In our example of TSP:

- Long term memory:
  - Maintain a list of t towns which have been considered in the last k best (worst) solutions
  - encourage (or discourage) their selections in future solutions
  - using their frequency of appearance in the set of elite solutions and the quality of solutions which they have appeared in our selection function

Example: TSP using Tabu Search

In our example of TSP:

- Aspiration:
  - If the next moves consider those moves in the tabu list but generate better solution than the current one
  - Accept that solution anyway
  - Put it into tabu list

Tabu Search Pros & Cons

- Pros
  - Generated generally good solutions for optimisation problems compared with other AI methods
- Cons
  - Tabu list construction is problem specific
  - No guarantee of global optimal solutions

Summary

- Use of memory in search
- Tabu Search algorithm
- Elements of Tabu Search
- Example of Tabu Search on TSP

References

- Glover, F. 1989. *Tabu Search – Part I*. ORSA Journal on Computing, Vol. 1, No. 3, pp 190-206.
- Glover, F. 1990. *Tabu Search – Part II*. ORSA Journal on Computing, Vol. 2, No. 1, pp 4-32.
- Glover, F., Laguna, M. 1998. *Tabu Search*. Kluwer Academic Publishers
- Rayward-Smith, V.J., Osman, I.H., Reeves, C.R., Smith, G.D. 1996. *Modern Heuristic Search Methods*, John Wiley & Sons.
- Russell, S., Norvig, P. 1995. *Artificial Intelligence A Modern Approach*. Prentice-Hall

G5BAIM
Artificial Intelligence Methods

Dr. Rong Qu

End of Tabu Search

SA vs. TS

	SA	TS
No. of neighbourhoods considered at each move	1	n
Accept worse moves? How?	Yes P = exp^(-c/t)	Yes The best neighbourhood if it is not tabu-ed.
Accept better moves?	Always	Always (aspiration)
Stopping conditions	T = 0, or at a low temperature, or No Improvement after certain number of iterations	Certain number of iterations, Or No improvement after certain number of iterations