# 6. Tabu Search
## 6.1 Basic Concepts

**Fall 2010**

*Instructor: Dr. Masoud Yaghini*

## Outline

# Introduction

## Introduction

- Tabu Search (TS)
  - a metaheuristic method proposed by Fred Glover
  - allows Local Search (LS) methods to overcome local optima.
- The basic principle of TS
  - pursuing LS whenever it encounters a local optimum by allowing non-improving moves
  - cycling back to previously visited solutions is prevented by the use of memories, called tabu lists, that record the recent history of the search

## Introduction

- TS is a general strategy for guiding and controlling "inner" heuristics specifically tailored to the problems at hand.
- TS can be seen as simply the combination of LS with short-term memories.
- TS can be applied directly to many kinds of decision problems, without the need to transform them into mathematical formulations.

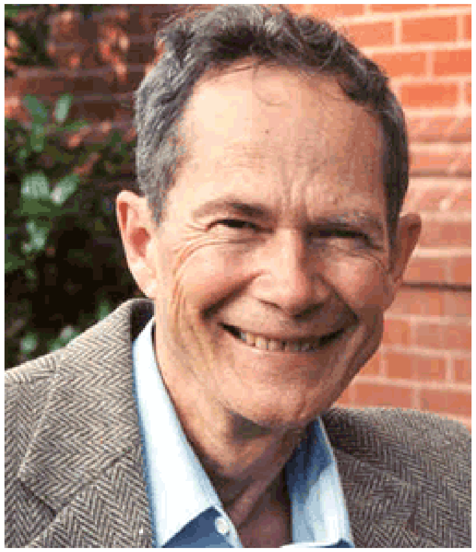
## History

- A very simple memory mechanism is described in Glover (1977) to implement the *oscillating assignment heuristic*

Glover, F. (1977) “Heuristics for Integer Programming Using Surrogate Constraints,” *Decision Sciences*, vol. 8, no. 1, pp. 156-166.

## History

- Glover (1986) introduces *tabu search* as a "meta-heuristic" superimposed on another heuristic



Glover, F. (1986) "Future Paths for Integer Programming and Links to Artificial Intelligence," *Computer and Operations Research*, vol. 13, no. 5, pp. 533-549.

## History

- Glover (1989a) and (1989b) provide a full description of the method

Glover, F. (1989a) "Tabu Search – Part I," *INFORMS Journal on Computing*, vol. 1, no. 3, pp. 190-206.
Glover, F. (1989b) "Tabu Search – Part II," *INFORMS Journal on Computing*, vol. 2, no. 1, pp. 4-32.

## The Word Tabu

- The word tabu (or taboo)
  - a language of Polynesia,
  - comes from Tongan people
  - where it was used by the people of Tonga island to indicate things that cannot be touched because they are sacred.
- According to Webster's Dictionary, the word tabu means
  - “a prohibition imposed by social custom as a protective measure" or
  - “something banned as constituting a risk.”

## The Word Tabu

- Tabus are transmitted by means of a **social memory** which is subject to **modification over time**.
  - This creates the fundamental link to the meaning of "tabu" in tabu search.
- Tabus in tabu search receive their status by reliance on an **evolving memory**, which allows this status to shift according to **time** and **circumstance**.

## Adaptive Memory

- Adaptive memory
  - capable of searching the solution space economically and effectively.
- TS contrasts with memoryless algorithms that heavily rely on semirandom processes
  - Examples of memoryless methods include the genetic algorithm and simulated annealing approaches.
- TS also contrasts with rigid memory algorithms typical of branch and bound methods.

## TS Components

- Tabu search components:
  - Search Space
  - Neighborhood Structure
  - Tabus
  - Aspiration Criteria
  - Termination Criteria
  - Candidate List Strategies
  - Intensification
  - Diversification

## Applications of TS

- **Scheduling**
  - Flow-Time Cell Manufacturing, Heterogeneous, Processor Scheduling, Workforce Planning, Classroom Scheduling, Machine Scheduling, Flow Shop Scheduling, Job Shop Scheduling, Sequencing and Batching
- **Telecommunications**
  - Call Routing, Bandwidth Packing, Hub Facility Location, Path Assignment, Network Design for Services, Customer Discount Planning, Failure Immune Architecture, Synchronous Optical Networks
- **Design**
  - Computer-Aided Design, Fault Tolerant Networks, Transport Network Design, Architectural Space Planning, Diagram Coherency, Fixed Charge Network Design, Irregular Cutting Problems

## Applications of TS

- **Production, Inventory and Investment**
  - Flexible Manufacturing, Just-in-Time Production, Capacitated MRP, Part Selection, Multi-item Inventory Planning, Volume Discount Acquisition, Fixed Mix Investment
- **Location and Allocation**
  - Multicommodity Location/Allocation, Quadratic Assignment, Quadratic Semi-Assignment, Multilevel Generalized Assignment, Lay-Out Planning, Off-Shore Oil Exploration
- **Routing**
  - Vehicle Routing, Capacitated Routing, Time Window Routing, Multi-Mode Routing, Mixed Fleet Routing, Traveling Salesman, Traveling Purchaser

## Applications of TS

- **Logic and Artificial Intelligence**
  - Maximum Satisfiability, Probabilistic Logic, Clustering
  - Pattern Recognition/Classification, Data Integrity, Neural Network Training and Design
- **Graph Optimization**
  - Graph Partitioning, Graph Coloring, Clique Partitioning, Maximum Clique Problems, Maximum Planner Graphs, P-Median Problems
- **Technology**
  - Seismic Inversion, Electrical Power Distribution, Engineering Structural Design, Minimum Volume Ellipsoids, Space Station Construction, Circuit Cell Placement

## Applications of TS

- **General Combinational Optimization**
  - Zero-One Programming, Fixed Charge Optimization, Nonconvex Nonlinear Programming, All-or-None Networks, Bilevel Programming, General Mixed Integer Optimization

## Sample Problems

## The Classical Vehicle Routing Problem

- Vehicle Routing Problems have important applications in the area of distribution management.
- They have become some of the most studied problems in the combinatorial optimization.
- These include several TS implementations that currently rank among the most effective.
- The Classical Vehicle Routing Problem (CVRP) is the basic variant in that class of problems.

## The Classical Vehicle Routing Problem

- The CVRP can formally be defined as follows:
  - G = (V, A) be a graph where **V** is the vertex set and **A** is the arc set, one of the vertices represents the depot and the other vertices customers that need to be serviced.
  - m : index of vehicles
  - Q : the capacity of vehicles
  - $v_i$ : the vertex of customer i
  - $q_i$ : the demand of customer i
  - $t_i$ : the service time of customer i
  - $c_{ij}$ : the cost of arc $(v_i, v_j)$ of A
  - $t_{ij}$ : the travel time of arc $(v_i, v_j)$ of A
  - L: maximum duration time of each route

## The Classical Vehicle Routing Problem

- The CVRP consists in finding a set of routes such that:
  - Each route begins and ends at the depot
  - Each customer is visited exactly once by exactly one route
  - The total demand of the customers assigned to each route does not exceed $Q$
  - The total duration of each route (including travel and service times) does not exceed a specified value $L$
  - The total cost of the routes is minimized

## The Capacitated Plant Location Problem

- The Capacitated Plant Location Problem (CPLP) is one of the basic problems in Location Theory.
- It is encountered in many application settings that involve locating facilities with limited capacity to provide services.

## The Capacitated Plant Location Problem

- The CPLP can be formally described as follows:
  - I : a set of customers
  - J : the set of potential sites
  - i : the index of customers
  - j : the index of sites
  - $d_i$ : the demand of customer $i \in I$, for some product are to be served from plants located in a subset of sites from a given set *J*
  - $f_i$ : the fixed cost of opening the plant at j
  - $K_j$ : the capacity of plant at j
  - $c_{ij}$: The cost of transporting one unit of the product from site j to customer i

## The Capacitated Plant Location Problem

- The CPLP (cont.)
  - $x_{ij}$: the quantity shipped from site j to customer i (flow or allocation variables)
  - $y_j$ : a 0–1 variable indicating whether or not the plant at site is open (location variables)
- The objective is to minimize the total cost, i.e., the sum of the fixed costs for open plants and the transportation costs.

## The Capacitated Plant Location Problem

- The mathematical program:

$$(CPLP) \quad \textit{Minimize } z = \sum_{j \in J} f_j y_j + \sum_{i \in I} \sum_{j \in J} c_{ij} x_{ij}$$

$$\textit{subject to} \sum_{j \in J} x_{ij} = di, \quad i \in I$$

$$\sum_{i \in I} x_{ij} \le K_j y_j, \quad j \in J$$

$$x_{ij} \ge 0, \quad i \in I, \; j \in J$$

$$y_j \in \{0, 1\}, \quad j \in J$$

## The Capacitated Plant Location Problem

- **Remark 1.** For any vector $\tilde{y}$ of location variables, optimal (to this plant configuration) values for the flow variables x($\tilde{y}$) can be retrieved by solving the associated transportation problem:

$$
\begin{aligned}
(TP) \quad & \textit{Minimize } z(\tilde{y}) = \sum_{i \in I} \sum_{j \in J} c_{ij} x_{ij} \\
& \textit{subject to } \sum_{j \in J} x_{ij} = d_i, \quad i \in I \\
& \sum_{i \in I} x_{ij} \leq K_j \tilde{y}_j, \quad j \in J \\
& x_{ij} \geq 0, \quad i \in I, \quad j \in J
\end{aligned}
$$

## The Capacitated Plant Location Problem

- If $\tilde{y} = y^*$, the optimal location variable vector, the optimal solution to the original CPLP problem is simply given by (**y***, **x(y*)**).

## The Capacitated Plant Location Problem

- **Remark 2.** An optimal solution of the original CPLP problem can always be found at an extreme point of the polyhedron of feasible flow vectors defined by the constraints:

$$\sum_{j \in J} x_{ij} = d_i, \quad i \in I$$

$$\sum_{i \in I} x_{ij} \le K_j, \quad j \in J$$

$$x_{ij} \ge 0, \quad i \in I, \ j \in J$$

## The Capacitated Plant Location Problem

- The CPLP can be interpreted as a fixed-charge problem defined in the space of the flow variables.
- This fixed-charge problem has a concave objective function that always admits an extreme point minimum.
- The optimal values for the location variables can easily be obtained from the optimal flow vector by setting equal to 1 if and to 0 otherwise.

## Search Space

## Search Space

- Search Space
  - The one basic elements of TS is the definition of Search Space
  - The search space is the space of all possible solutions that can be considered (visited) during the search.

## Example: Search space in the CVRP

- **Search space in the CVRP** is the set of feasible solutions to the problem
- Each point in the search space corresponds to a set of vehicles routes satisfying all the specified constraints

## Example: Search space in the CPLP

- In the **CPLP** the feasible space involves both integer location variables and continuous flow variables that are linked by strict conditions
- Option 1:
  - the full feasible space
  - this would involve manipulating both location and flow variables
- Option 2:
  - the set of feasible vectors of location variables
  - any solution in this space being completed to yield a feasible solution by computing the associated optimal flow variables.

## Example: Search space in the CPLP

- Option 3:
  - on the basis of Remark 2, one could also decide to search instead the set of extreme points of the set of **feasible flow vectors**
  - retrieving the associated location variables by simply noting that a plant must be open whenever some flow is allocated to it.

## Search Space

- It is also important to note that it is not always a good idea to restrict the search space to feasible solutions
- In many cases, allowing the search to move to infeasible solutions is desirable, and sometimes necessary.

## Neighborhood Structure

## Neighborhood Structure

- **Neighboring Solutions**
  - The another basic element of TS is the definition of **Neighboring Solutions.**
- At each iteration, the local transformations that can be applied to the current solution, denoted ***S***, define a set of neighboring solutions in the search space, denoted ***N(S)*** (the neighborhood of S).
- Formally, ***N(S)*** is a subset of the search space defined by:

  ***N(S)*** = {solutions obtained by applying a single local transformation (move) to *S*}.
- For any specific problem there are many possible neighborhood structures.

## Move

- Tabu search begins in the same way as ordinary **local search or neighborhood search**, proceeding iteratively from one point (solution) to another until a chosen **termination criterion** is satisfied.
- Move
  - Each solution has an associated neighborhood ***N(S)***, and each neighboring solutions is reached from current solution by an operation called a **move**.

## Example: Neighborhood Structure in CVRP

- Simple neighborhood structures for the CVRP involve:
  - A single customer is inserted in the same route, or
  - A single customer is inserted in another route with sufficient residual capacity.

## Example: Neighborhood Structure in CVRP

- The neighborhood structures is the way in which insertions are performed:
- Method 1:
  - one could use random insertion
- Method 2:
  - insertion at the best position in the target route
- Method 3:
  - one could use more complex insertion schemes that involve a partial re-optimization of the target route.
- Method 4:
  - More complex neighborhood structures for the CVRP are obtained by allowing simultaneously the movement of customers to different routes and the swapping of customers between routes.

## Example: Neighborhood Structure in CPLP

- When different definitions of the search space are considered for any given problem, neighborhood structures will inevitably differ to a considerable degree.
- This can be illustrated on the CPLP example.

## Example: Neighborhood Structure in CPLP

- If the search space is defined with respect to the location variables:
  - neighborhood structures will usually involve the so-called "Add/Drop" and "Swap" moves that respectively change the status of one site
    - i.e., either opening a closed facility or closing an open one
  - move an open facility from one site to another (this move amounts to performing simultaneously an Add move and a Drop move.

## Example: Neighborhood Structure in CPLP

- If the search space is the **set of extreme points of the set of feasible flow vectors**:
  - moves can be defined by the application of pivots to the linear programming formulation of the transportation problem
  - since each pivot operation moves the current solution to an adjacent extreme point

## Neighborhood Structure

- Choosing a search space and a neighborhood structure is the most critical step in the design of any TS metaheuristic.
- At this step that one must make the best use of the understanding and knowledge of the problem at hand.

## Tabus

## Tabus

- Tabus
  - are used to prevent cycling when **moving away** from local optima
  - They prevent the search from tracing back its steps to where it came from.
  - This is achieved by declaring tabu or disallowing moves
  - Tabus are also useful to help the search move away from previously visited portions of the search space and thus perform more extensive exploration.

## Tabus

- Tabu tenure
  - the number of iterations that a move is declared being in tabu
  - If *tabu_tenure* too small -> risk of cycling
  - If *tabu_tenure* too large -> may restrict the search too much

## Tabus

- **Short-term Memory / Tabu List**
  - Tabus are stored in a **short-term memory** of the search (or **tabu list**) and usually only a fixed and fairly limited quantity of information is recorded.

## Tabus

- Possibilities regarding the specific information that is recorded in tabu lists:
  - First: recording complete solutions
    - this requires a lot of storage and makes it expensive to check whether a potential move is tabu or not
    - it is therefore seldom used.
  - Second: recording the last few solutions
    - The most commonly used tabus
    - Involve recording the last few solutions performed on the current solution and prohibiting reverse transformations
  - Third: recording the moves
    - Some are based on key characteristics of the solutions themselves or of the **moves**.

## Tabus in CVRP

- In the CVRP, if customer $v_1$ has just been moved from $R_1$ route to $R_2$ route
- Tabus can be defined in several ways.
- Method 1:
  - one could declare tabu specifically moving back $v_1$ from $R_2$ to $R_1$ and record this in the short-term memory as the triplet ($v_1$, $R_2$, $R_1$)
  - this type of tabu will not constrain the search much, but that cycling may occur if $v_1$ is then moved to another route $R_3$ and then from $R_3$ to $R_1$.

## Tabus in CVRP

- Method 2:
  - A stronger tabu would involve prohibiting moving back $v_1$ to $R_1$ (without consideration for its current route) and be recorded as $(v_1, R_1)$.
- Method 3:
  - An even stronger tabu would be to disallow moving $v_1$ to any other route and would simply be noted as $(v_1)$,

## Tabus in CPLP

- When searching the space of location variables:
  - Tabus for Add/Drop moves
    - tabus on Add/Drop moves should prohibit changing the status of the affected location variable and can be recorded by noting its index.
  - Tabus for Swap moves
    - Tabus for Swap moves are more complex
    - they could be declared with respect to the site where the facility was closed, to the site where the facility was opened, to both locations (i.e., changing the status of both location variables is tabu), or to the specific swapping operation.

## Tabus in CPLP

- When searching the space of flow variables:
  - one can take advantage of the fact that a pivot operation is associated with a unique pair of entering and leaving variables to define tabus
  - while here again several combinations are possible, experience has shown that when dealing with pivot neighborhood structures, tabus imposed on leaving variables (to prevent them from coming back in the basis) are usually much more effective.

## Multiple Tabu Lists

- Multiple tabu lists can be used simultaneously and are sometimes advisable.
- For instance, in the CPLP:
  - if one uses a neighborhood structure that contains both Add/Drop and Swap moves
  - it might be a good idea to keep a separate tabu list for each type of moves.

## Varying the Tabu List

- Standard tabu lists are usually implemented as circular lists of fixed length.
- It has been shown, however, that fixed-length tabus cannot always prevent cycling, and some authors have proposed varying the tabu list length during the search
- Another solution is to randomly generate the tabu tenure of each move within some specified interval

## Aspiration Criteria

## Aspiration Criteria

- Tabus are sometimes too powerful:
  - they may prohibit attractive moves, even when there is no danger of cycling, or
  - they may lead to an overall stagnation of the searching process.
- It is thus necessary to use algorithmic devices that will allow one to revoke (cancel) tabus.
- These are called aspiration criteria.

## Aspiration Criteria

- **Most commonly used aspiration criterion**
  - allowing a move even if it is tabu and if it results in a solution with an objective value better than that of the current best-known solution
  - the new solution has obviously not been previously visited
- The key rule in this respect is that if cycling cannot occur, tabus can be disregarded.

## Termination Criteria

## Termination Criteria

- In theory, the search could go on forever, unless the optimal value of the problem at hand is known beforehand.
- In practice, obviously, the search has to be stopped at some point.

## Termination Criteria

- **Most commonly used stopping criteria**:
  - after a fixed number of iterations
  - after a fixed amount of CPU time
  - after some number of iterations without an improvement in the objective function value (the criterion used in most implementations)
  - when the objective reaches a pre-specified threshold value
- In **complex tabu schemes**, the search is usually stopped after completing a sequence of phases, the duration of each phase being determined by one of the above criteria.

## Candidate List Strategies

## Candidate List Strategies

- In regular TS, one must evaluate the objective for every element of the neighborhood *N(S)* of the current solution.
- This can prove extremely expensive from the computational standpoint.
- An alternative is to control the number of moves examined is by means of **candidate list strategies**, which provide the strategic ways of generating a useful subset *N'(S)* of *N(S)*.

## Candidate List Strategies

- The probabilistic approach can be considered to be one instance of a candidate list strategy
- It is to instead consider only a random sample $N'(S)$ of $N(S)$, thus reducing significantly the computational burden.
- One the negative side, it must be noted that, in that case, one may miss excellent solutions

## A Template for Simple Tabu Search

## A Template for Simple Tabu Search

- We suppose that we are trying to minimize a function *f(S)* and we choose at each iteration the **best available move**.
- Notation
  - S : the current solution
  - S* : the best-known solution
  - f* : the value of S*
  - N(S) : the neighborhood of *S*
  - Ñ(S) *:* the admissible subset of *N(S)* , i.e., non-tabu or allowed by aspiration.
  - T : the tabu list

## A Template for Simple Tabu Search

- **Initialization**
  - Construct an initial solution $S_0$.
  - Set $S := S_0$, $f^* := f(S_0)$, $S^* := S_0$, $T := \emptyset$
- **Search**
  - While termination criterion not satisfied do
    - Select $S$ in $\operatorname{argmin}[f(S')]$; $S' \in \tilde{N}(S)$
    - If $f(S) < f^*$, then set $f^* := f(S)$, $S^* := S$
    - Update tabu list $T$
      - record tabu for the current move in $T$;
      - delete oldest entry if necessary;
  - Endwhile.

## References

## References

- F. Glover and M. Laguna, "**Tabu Search**," In Handbook of Applied Optimization, P. M. Pardalos and M. G. C. Resende (Eds.), Oxford University Press, pp. 194-208 (2002).
- Gendreau M. **An Introduction to Tabu Search**. In Handbook of Metaheuristics, Kochenberger G, Glover F, (Editors), Dordrecht, Kluwer Academic Publishers, 2003.

## The End