Tabu Search

Subset of Slides from

Lei Li, HongRui Liu, Roberto Lu

Introduction

- Glover, F. 1986. Future Paths for Integer Programming and Links to Artificial Intelligence. *Computers and Operations Research.* Vol. 13, pp. 533-549.

- Hansen, P. 1986. The Steepest Ascent Mildest Descent Heuristic for Combinatorial Programming. *Congress on Numerical Methods in Combinatorial Optimization*, Capri, Italy.

Tabu Search Strategy

- 3 main strategies [7]:
  - **Forbidding strategy**: control what enters the tabu list
  - **Freeing strategy**: control what exits the tabu list and when
  - **Short-term strategy**: manage interplay between the forbidding strategy and freeing strategy to select trial solutions

Parameters of Tabu Search [5]

- Local search procedure
- Neighborhood structure
- Aspiration conditions
- Form of tabu moves
- Addition of a tabu move
- Maximum size of tabu list
- Stopping rule

Basic Ingredients of Tabu Search

- A chief way to exploit memory in tabu search is to classify a subset of the moves in a neighborhood as forbidden (or ***tabu***) [1].
- A ***neighborhood*** is constructed to identify adjacent solutions that can be reached from current solution [8].
- The classification depends on the history of the search, and particularly on the recency or frequency that certain move or solution components, called ***attributes***, have participated in generating past solutions [1].
- A ***tabu list*** records forbidden moves, which are referred to as ***tabu moves*** [5].
- Tabu restrictions are subject to an important exception. When a tabu move has a sufficiently attractive evaluation where it would result in a solution better than any visited so far, then its tabu classification may be overridden. A condition that allows such an override to occur is called an ***aspiration criterion***[1].

Basic Tabu Search Algorithm [4]

- Step 1: Choose an initial solution i in S. Set i* = i and k=0.
- Step 2: Set k=k+1 and generate a subset V* of solution in N(i,k) such that either one of the Tabu conditions is violated or at least one of the aspiration conditions holds.
- Step 3: Choose a best j in V* and set i=j.
- Step 4: If f(i) < f(i*) then set i* = i.
- Step 5: Update Tabu and aspiration conditions.
- Step 6: If a stopping condition is met then stop. Else go to Step 2.

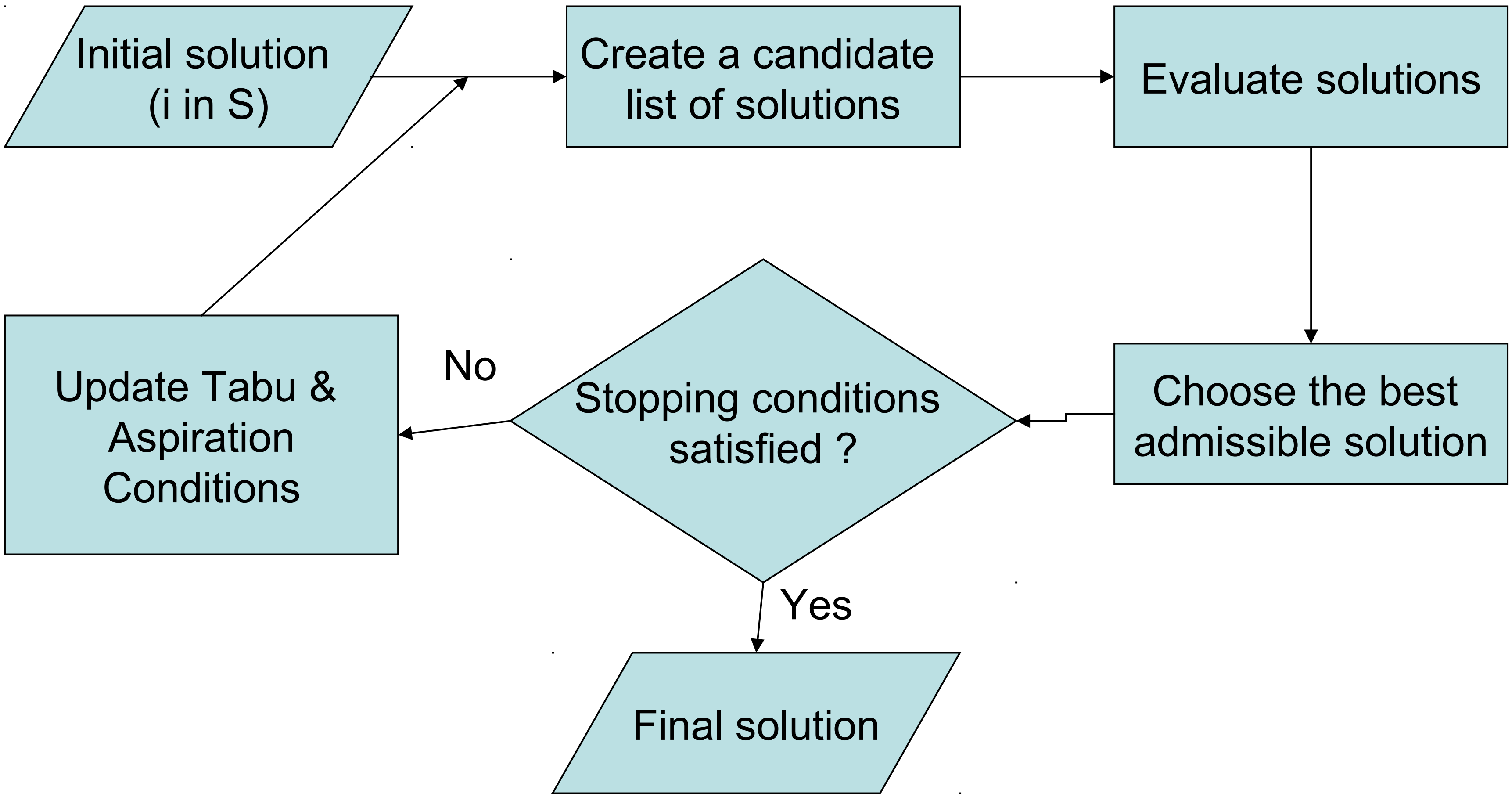
Tabu Search Stopping Conditions

Some immediate stopping conditions could be the following [4]:

1. N(i, K+1) = 0. (no feasible solution in the neighborhood of solution i)
2. K is larger than the maximum number of iterations allowed.
3. The number of iterations since the last improvement of i* is larger than a specified number.
4. Evidence can be given than an optimum solution has been obtained.

Hillier and Lieberman [5] outlined the tabu search stopping criterion by, for example, using a fixed number of iterations, a fixed amount of CPU time, or a fixed number of consecutive iterations without an improvement in the best objective function value. Also stop at any iteration where there are no feasible moves into the local neighborhood of the current trial solution.

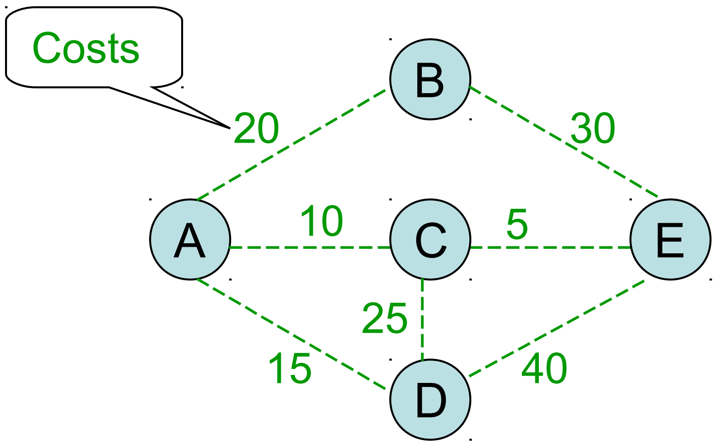
Flowchart of a Standard Tabu Search Algorithm [7]

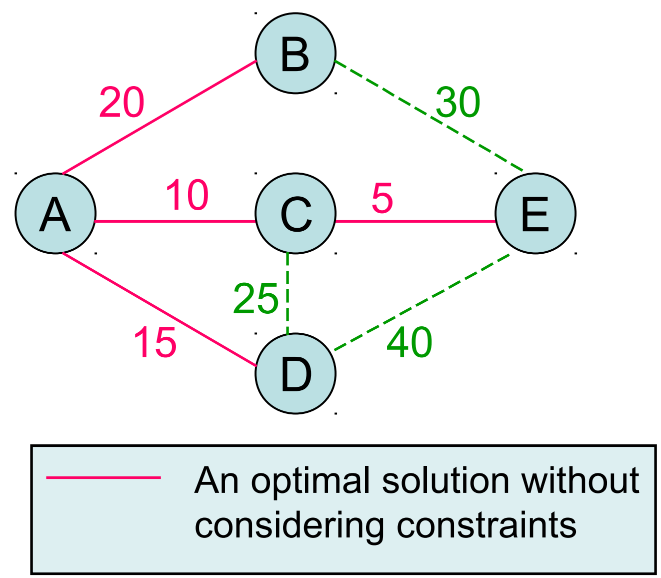


Example [5]

– Minimum spanning tree problem with constraints.

– *Objective: Connects all nodes with minimum costs*

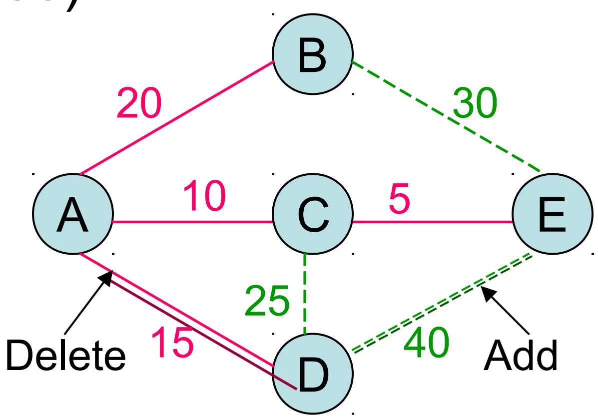




Constraints 1: Link AD can be included only if link DE also is included. (penalty:100)
Constraints 2: At most one of the three links – AD, CD, and AB – can be included.
(Penalty of 100 if selected two of the three, 200 if all three are selected.)

Example

Iteration 1
Cost=50+200 (constraint penalties)



Add	Delete	Cost
BE	CE	75+200=275
BE	AC	70+200=270
BE	AB	60+100=160
CD	AD	60+100=160
CD	AC	65+300=365
DE	CE	85+100=185
DE	AC	80+100=180
DE	AD	**75+0=75**

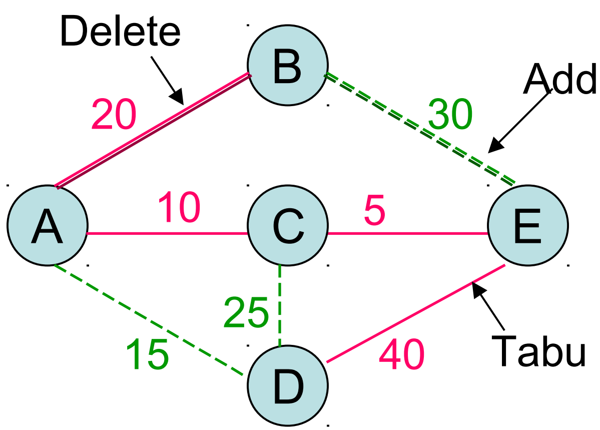
New cost = 75 (iteration 2)
(local optimum)

Constraints 1: Link AD can be included only if link DE also is included. (penalty:100)
Constraints 2: At most one of the three links – AD, CD, and AB – can be included.
(Penalty of 100 if selected two of the three, 200 if all three are selected.)

Example

Tabu list: DE

Iteration 2 Cost=75



Add	Delete	Cost
AD	DE*	Tabu move
AD	CE	85+100=185
AD	AC	80+100=180
BE	CE	100+0=100
BE	AC	95+0=95
BE	AB	**85+0=85**
CD	DE*	60+100=160
CD	CE	95+100=195

**** A tabu move will be considered only if it would result in a better solution than the best trial solution found previously (Aspiration Condition)***

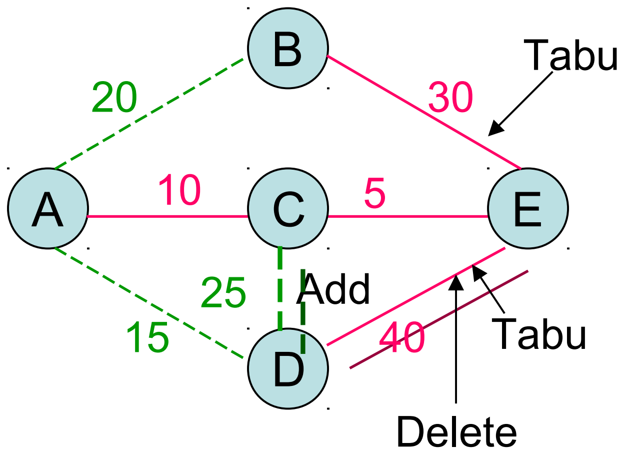
Iteration 3 new cost = 85 Escape local optimum

Constraints 1: Link AD can be included only if link DE also is included. (penalty:100)
Constraints 2: At most one of the three links – AD, CD, and AB – can be included. (Penalty of 100 if selected two of the three, 200 if all three are selected.)

Example

Tabu list: DE & BE

Iteration 3 Cost=85



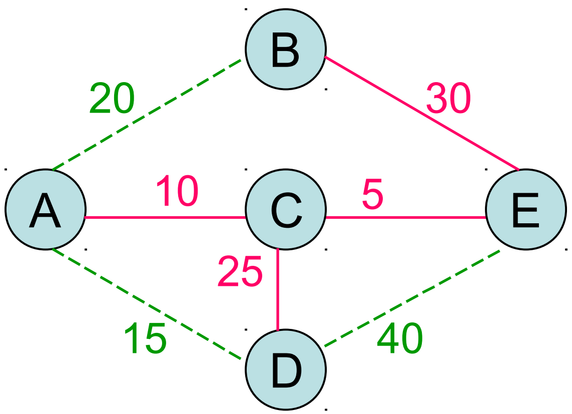
Add	Delete	Cost
AB	BE*	Tabu move
AB	CE	100+0=100
AB	AC	95+0=95
AD	DE*	60+100=160
AD	CE	95+0=95
AD	AC	90+0=90
CD	DE*	**70+0=70**
CD	CE	105+0=105

*** A tabu move will be considered only if it would result in a better solution than the best trial solution found previously (Aspiration Condition)***

Iteration 4 new cost = **70** Override tabu status

Constraints 1: Link AD can be included only if link DE also is included. (penalty:100)
Constraints 2: At most one of the three links – AD, CD, and AB – can be included. (Penalty of 100 if selected two of the three, 200 if all three are selected.)

Example



Optimal Solution

Cost = 70

Additional iterations only find inferior solutions

Pros and Cons

- Pros:
  - Allows non-improving solution to be accepted in order to escape from a local optimum
  - The use of Tabu list
  - Can be applied to both discrete and continuous solution spaces
  - For larger and more difficult problems (scheduling, quadratic assignment and vehicle routing), tabu search obtains solutions that rival and often surpass the best solutions previously found by other approaches [1].

- Cons:
  - Too many parameters to be determined
  - Number of iterations could be very large
  - Global optimum may not be found, depends on parameter settings

References

[1] Glover, F., Kelly, J. P., and Laguna, M. 1995. Genetic Algorithms and Tabu Search: Hybrids for Optimization. *Computers and Operations Research*. Vol. 22, No. 1, pp. 111 – 134.

[2] Glover, F. and Laguna, M. 1997. Tabu Search. Norwell, MA: Kluwer Academic Publishers.

[3] Hanafi, S. 2001. On the Convergence of Tabu Search. Journal of Heuristics. Vol. 7, pp. 47 – 58.

[4] Hertz, A., Taillard, E. and Werra, D. A Tutorial on Tabu Search. Accessed on April 14, 2005: http://www.cs.colostate.edu/~whitley/CS640/hertz92tutorial.pdf

[5] Hillier, F.S. and Lieberman, G.J. 2005. Introduction to Operations Research. New York, NY: McGraw-Hill. 8^{th} Ed.

[6] Ji, M. and Tang, H. 2004. Global Optimizations and Tabu Search Based on Mamory. *Applied Mathematics and Computation*. Vol. 159, pp. 449 – 457.

[7] Pham, D.T. and Karaboga, D. 2000. Intelligent Optimisation Techniques – Genetic Algorithms, Tabu Search, Simulated Annealing and Neural Networks. London: Springer-Verlag.

[8] Reeves, C.R. 1993. Modern Heuristic Techniques for Combinatorial Problems. John Wiley & Sons, Inc.