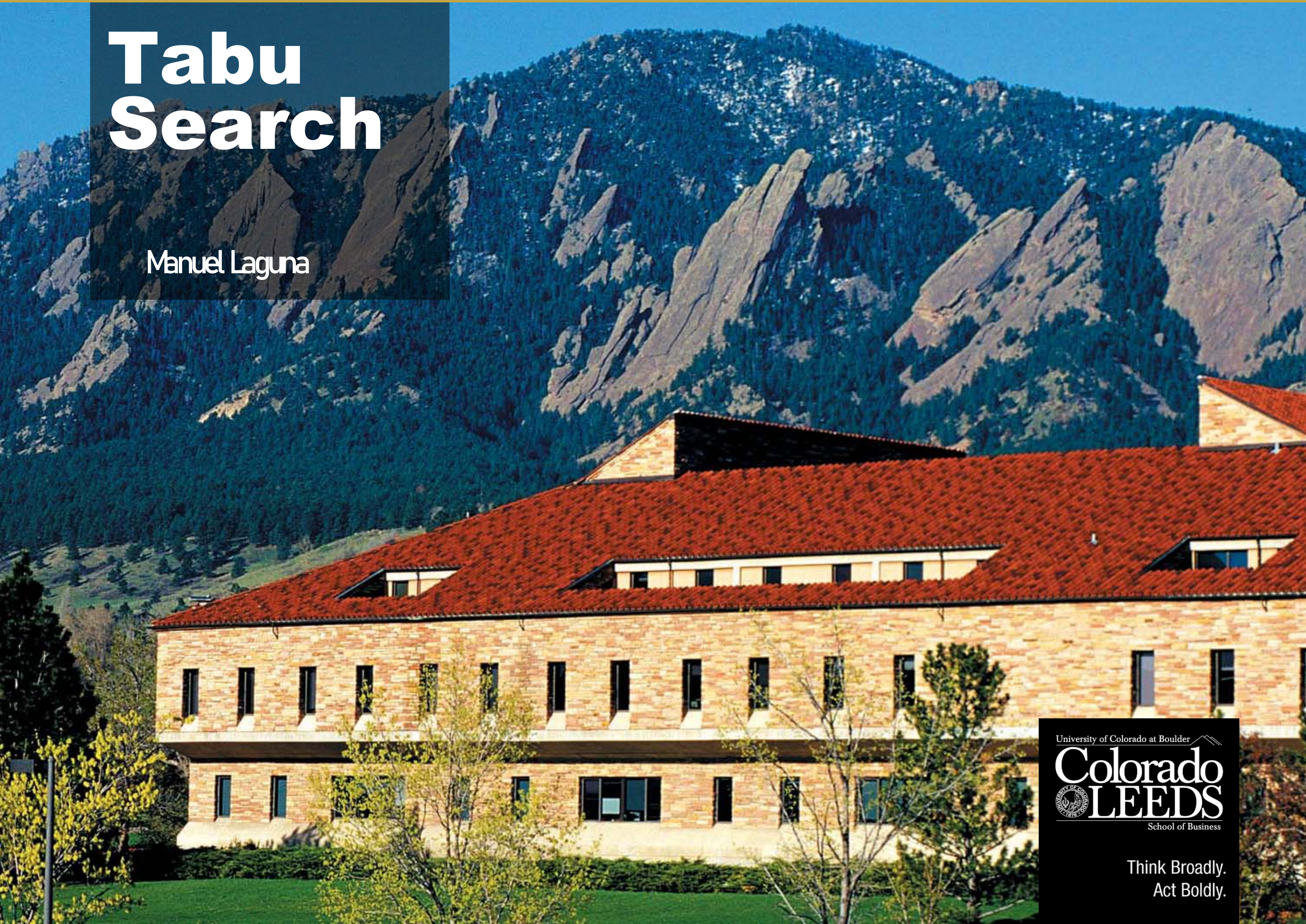
UNIVERSITY OF COLORADO AT BOULDER LEEDS SCHOOL OF BUSINESS

Tabu Search

Manuel Laguna



Outline

- Background
- Short Term Memory
- Long Term Memory
- Related Tabu Search Methods

Background

- Tabu search is a metaheuristic that guides a local search procedure to explore the solution space beyond local optimality
- Memory-based strategies are the hallmark of tabu search approaches

Basic Concepts

- Solution
  - Initial
  - Current
  - Best
- Move
  - Attributes
  - Value
- Neighborhood
  - Original
  - Modified (Reduced or Expanded)
- Tabu
  - Status
  - Activation rules

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

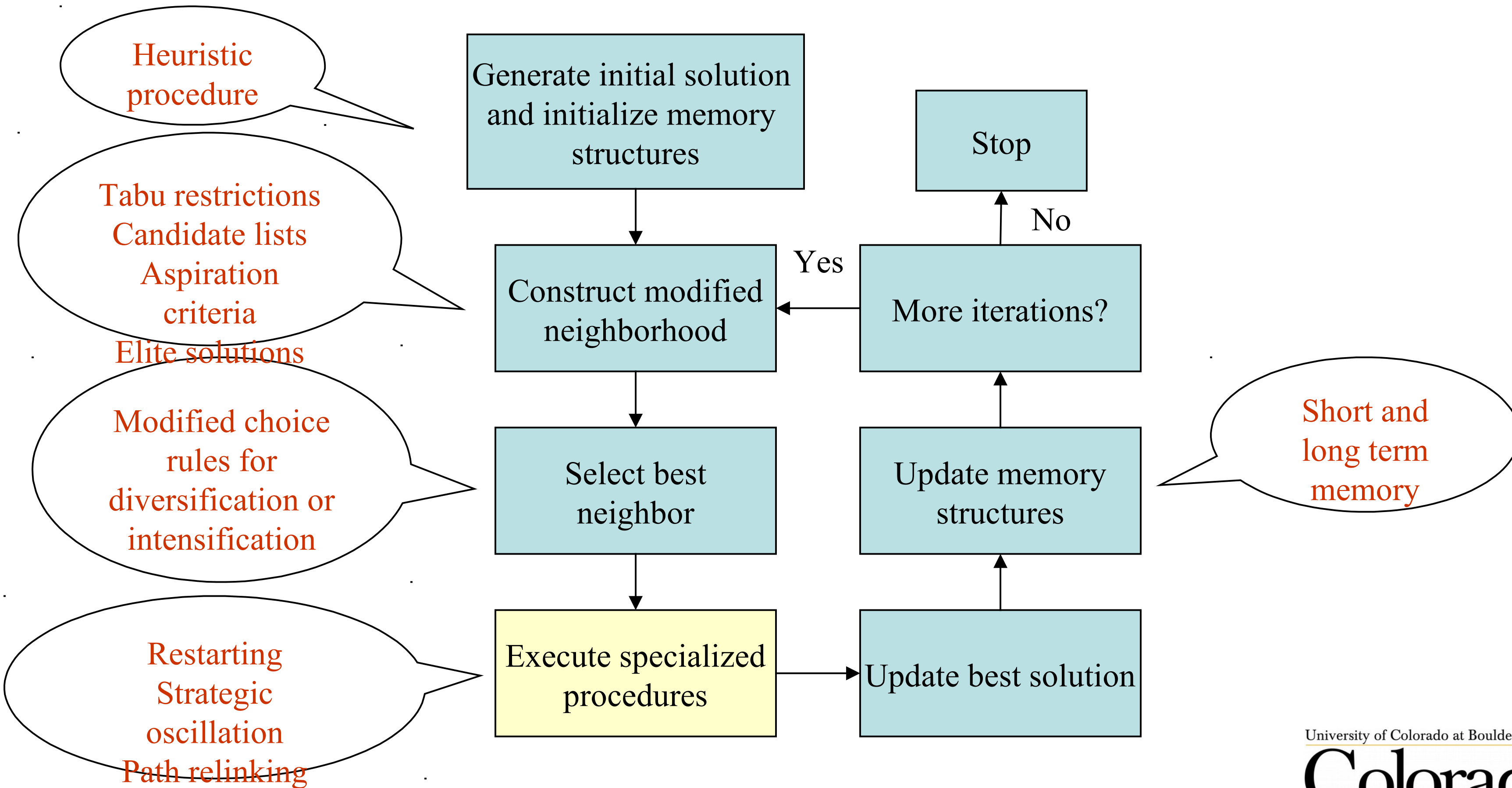
Glover, F. (1986) "Future Paths for Integer Programming and Links to Artificial Intelligence," *Computer and Operations Research*, vol. 13, no. 5, pp. 533-549.

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.

Tabu Search Framework



Short-Term Memory

- The main goal of the STM is to avoid reversal of moves and cycling

- The most common implementation of the STM is based on move attributes and the recency of the moves

Example 1

- After a move that changes the value of x_i from 0 to 1, we would like to prevent x_i from taking the value of 0 in the next $TabuTenure$ iterations
  - Attribute to record: i
  - Tabu activation rule: move ($x_i \leftarrow 0$) is tabu if i is tabu-active

Example 2

- After a move that exchanges the positions of element i and j in a sequence, we would like to prevent elements i and j from exchanging positions in the next $TabuTenure$ iterations

  – Attributes to record: *i* and *j*
  – Tabu activation rule: move ($i \leftrightarrow j$) is tabu if both *i* and *j* are tabu-active

Example 3

- After a move that drops element i from and adds element j to the current solution, we would like to prevent element i from being added to the solution in the next $TabuAddTenure$ iterations and prevent element j from being dropped from the solution in the next $TabuDropTenure$ iterations

  - Attributes to record: i and j
  - Tabu activation rules:
    - move (Add i) is tabu if i is tabu-active
    - move (Drop j) is tabu if j is tabu-active

Tabu or not Tabu

- Only moves can be tabu. Attributes are never tabu, they can only be tabu-active
- A move may be tabu if it contains one or more tabu-active attributes
- The classification of a move (as tabu or not tabu) is determined by the tabu-activation rules

TabuEnd Memory Structure

- This memory structure records the time (iteration number) when the tabu-active status of an attribute ends

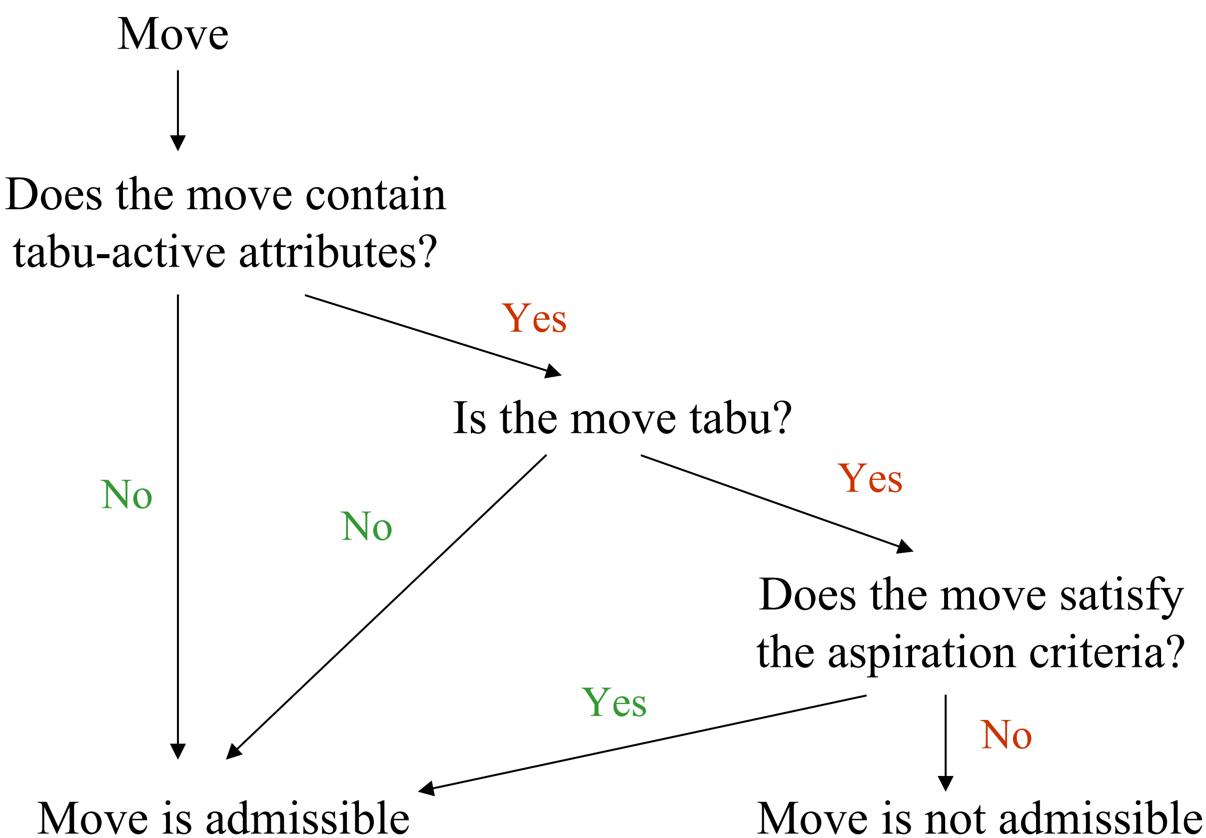
- Update after a move

  TabuEnd(Attribute) = Iter + TabuTenure

- Attribute is active if

  Iter $\leq$ TabuEnd(Attribute)

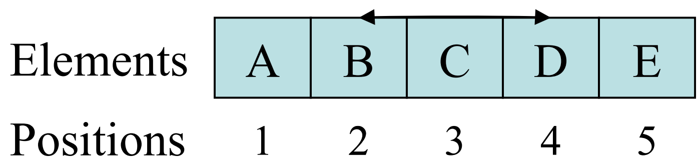
Tabu Decision Tree



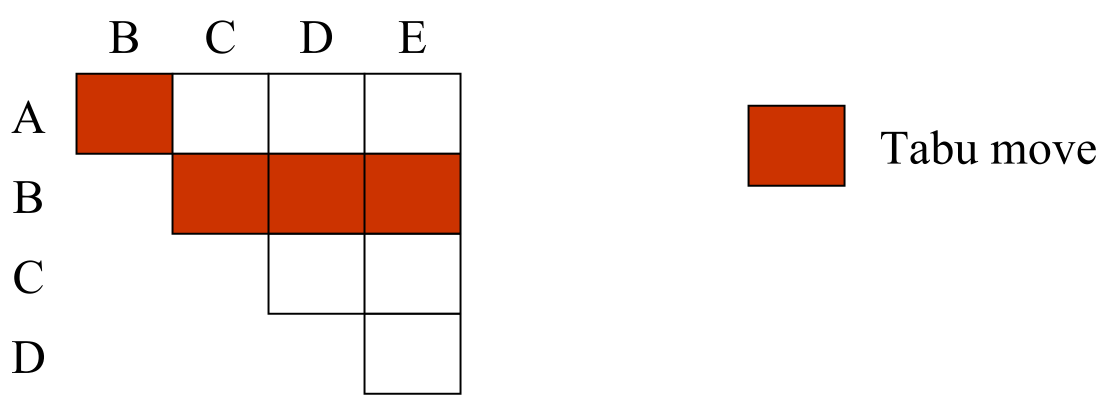
Search Flexibility

- The number of admissible moves in the neighborhood of the current solution depends on the …
  - Move type
  - Tabu activation rules
  - Tabu tenure
  - Aspiration criteria

Example 4



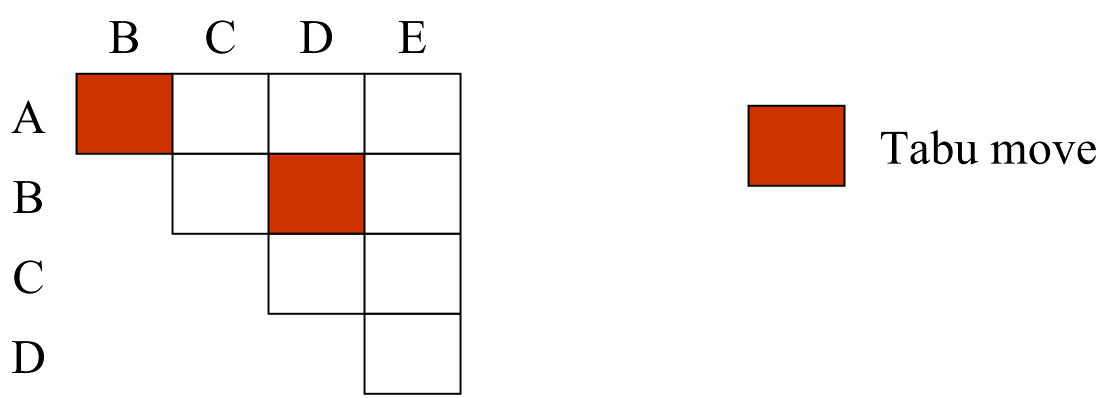
Tabu activation rule: move (B ↔ *) is tabu



Example 5

Elements	A	B	C	D	E
Positions	1	2	3	4	5

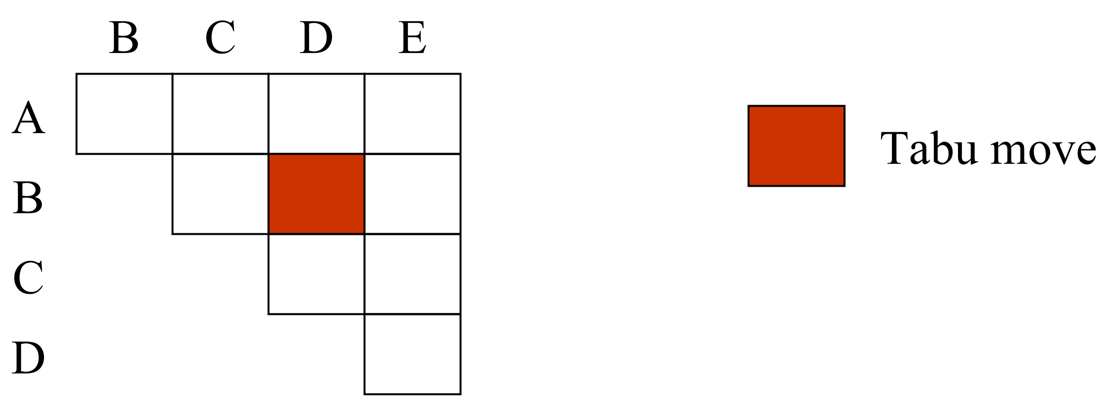
<u>Tabu activation rule</u>: move (B ↔ *) is tabu if B moves to 2 or earlier



Example 6

Elements	A	B	C	D	E
Positions	1	2	3	4	5

<u>Tabu activation rule</u>: move (B $\leftrightarrow$ D) is tabu



Tabu Tenure Management

- Static Memory
  - The value of *TabuTenure* is fixed and remains fixed during the entire search
  - All attributes remain tabu-active for the same number of iterations

- Dynamic Memory
  - The value of *TabuTenure* is not constant during the search
  - The length of the tabu-active status of attributes varies during the search

Simple Dynamic Tabu Tenure

- Update after a move

  TabuEnd(Attribute) = Iter + U(MinTenure, MaxTenure)

- The values of MinTenure and MaxTenure are search parameters

Aspiration Criteria

- By Objective
  - A tabu move becomes admissible if it yields a solution that is better than an aspiration value
- By Search Direction
  - A tabu move becomes admissible if the direction of the search (improving or non-improving) does not change

Candidate List Strategies

- Candidate lists are used to reduce the number of solutions examined on a given iterations

- They isolate regions of the neighborhood containing moves with desirable features

First Improving

- Choose the first improving move during the exploration of the current neighborhood

- This is a special case of the *Aspiration Plus* Candidate List Strategy
  - Threshold = Current Solution Value
  - Plus = 0
  - Min = 0
  - Max = Size of the neighborhood

Example 7

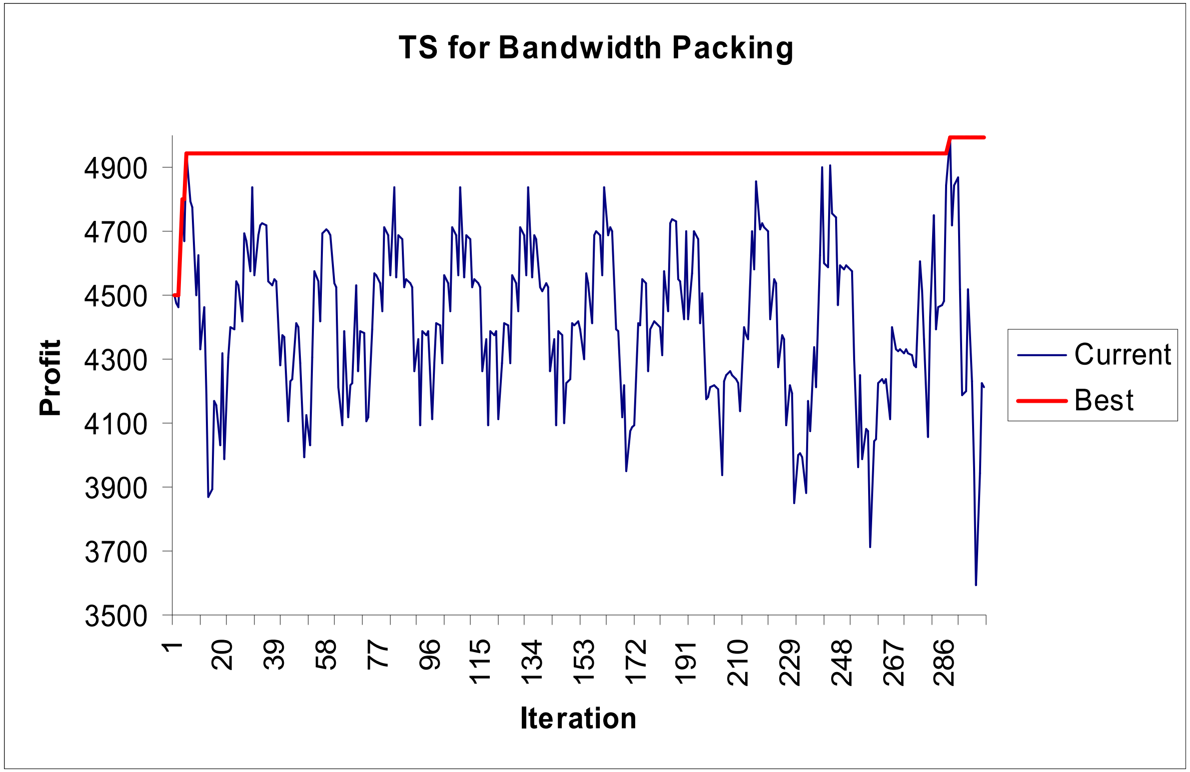
Move	Iteration 1	Iteration 2	Iteration 3	Iteration 4
1	NI(1)		NI(2)	NI(5)
2	NI(2)		NI(3)	NI(6)
3	NI(3)		NI(4)	NI(7)
4	I		NI(5)	NI(8)
5		NI(1)	NI(6)	NI(9)
6		NI(2)	I	NI(10)
7		NI(3)		NI(1)
8		NI(4)		NI(2)
9		I		NI(3)
10			NI(1)	NI(4)

Chosen move

Long Term Memory

- Frequency-based memory
- Strategic oscillation
- Path relinking

Effect of Long Term Memory



Frequency-based Memory

- Transition Measure
  - Number of iterations where an attribute has been changed (e.g., added or deleted from a solution)
- Residence Measure
  - Number of iterations where an attribute has stayed in a particular position (e.g., belonging to the current solution)

Example 8

- Transition Measure
  - Number of times that element i has been moved to an earlier position in the sequence sequence
- Residence Measure
  - Number of times that element i has occupied position k

Modifying Choice Rules

- Frequency-based memory is typically used to modify rules for …
  - choosing the best move to make on a given iteration
  - choosing the next element to add to a restarting solution
- The modification is based on penalty functions

Modifying Move Values for Diversification

Modified move value = Move value – Diversification parameter * F(frequency measure)

- Rule
  - Choose the move with the best move value if at least one admissible improving move exists
  - Otherwise, choose the admissible move with the best modified move value

Example 9

- The frequency of elements occupying certain positions can be used to bias a construction procedure and generate new restarting points

- For instance, due dates can be modified with frequency information (of jobs finishing on time) before reapplying the EDD rule

Strategic Oscillation

- Strategic oscillation operates by orienting moves in relation to a boundary

- Such an oscillation boundary often represents a point where the method would normally stop or turn around

Example 10

- In the knapsack problem, a TS may be designed to allow variables to be set to 1 even after reaching the feasibility boundary

- After a selected number of steps, the direction is reversed by choosing moves that change variables from 1 to 0

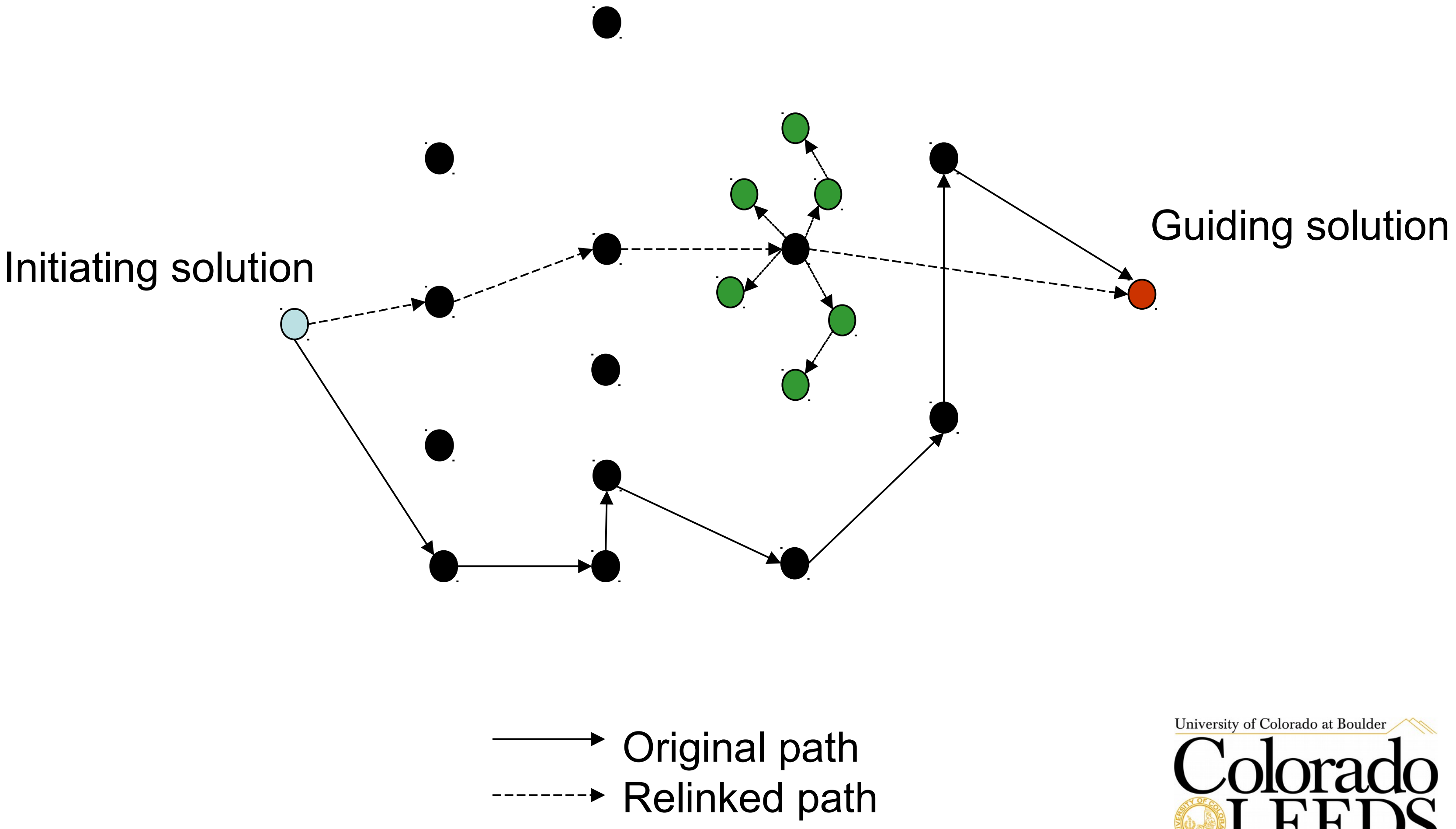
Example 11

- In the Min k-Tree problem, edges can be added beyond the critical level defined by k
- Then a rule is applied to delete edges
- Different rules would be typically used to add and delete edges

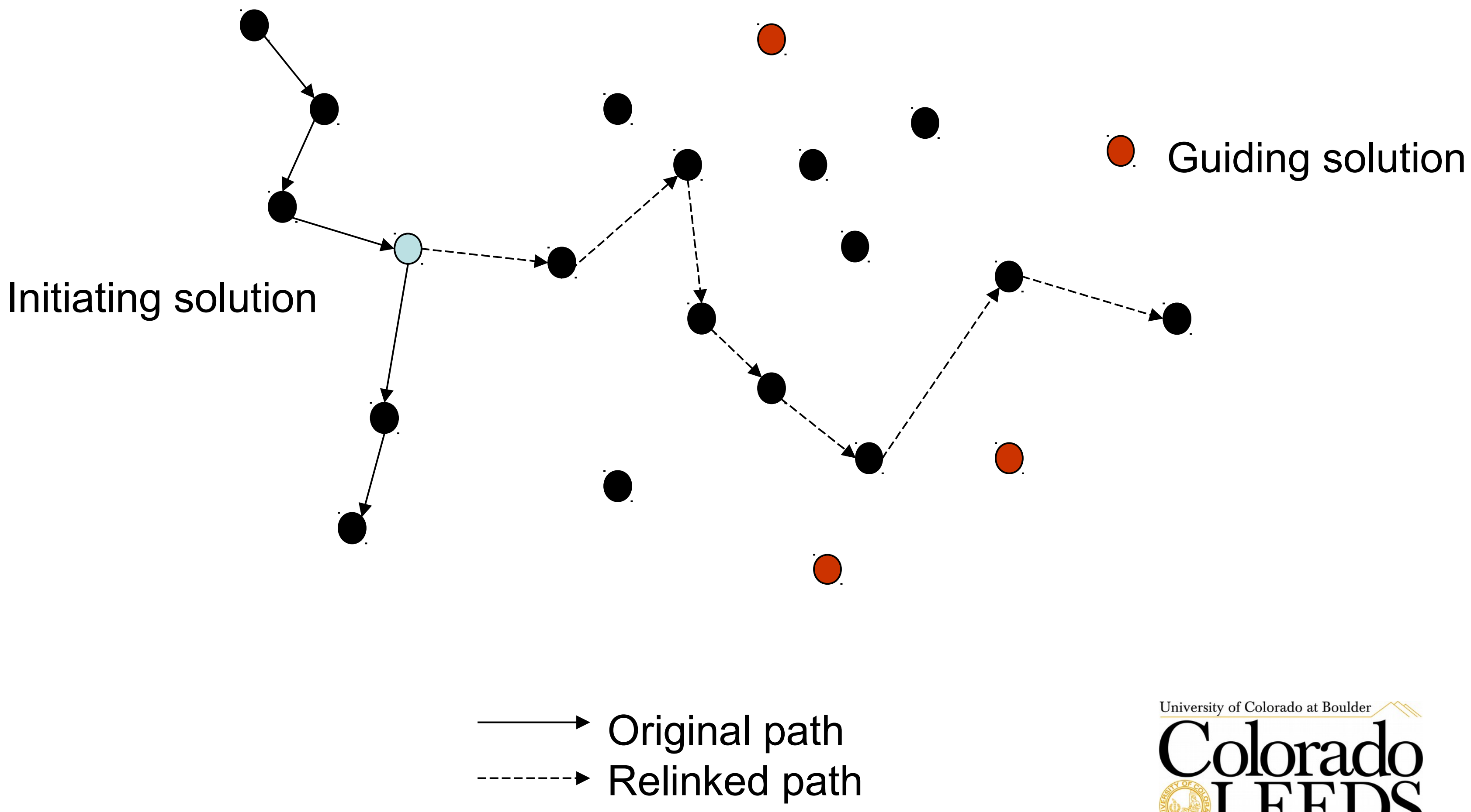
Path Relinking

- This approach generates new solutions by exploring trajectories that connect elite solutions
- The exploration starts from an initiating solution and generates a path in the neighborhood space that leads to a guiding solution
- Choice rules are designed to incorporate attributes contained in the guiding solution

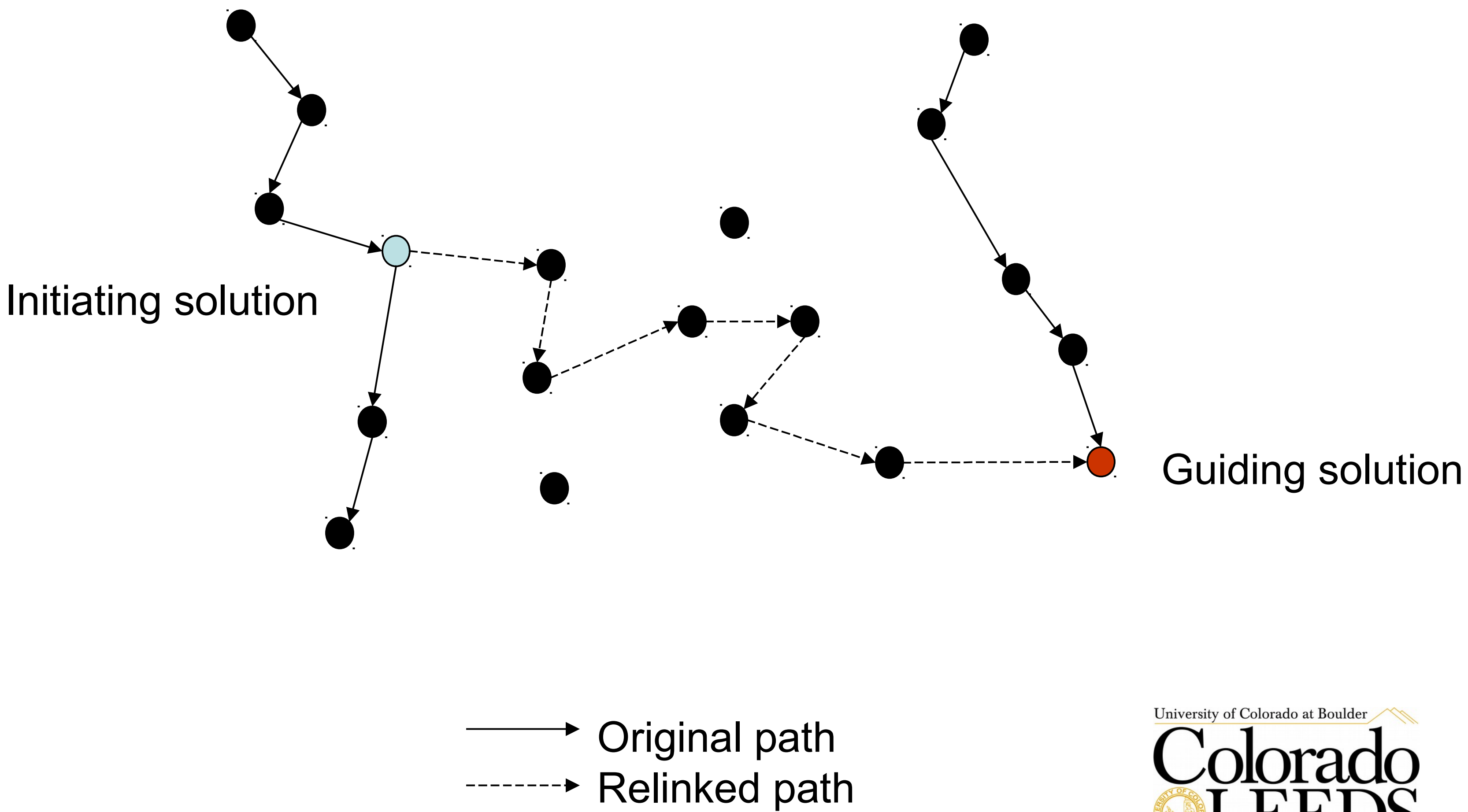
Relinking Solutions



Multiple Guiding Solutions



Linking Solutions



GRASP with Path Relinking

- Originally suggested in the context of Graph Drawing by Laguna and Martí (1999)

- Extensions and a comprehensive review are due to Resende and Riberio (2003) “GRASP with Path Relinking: Recent Advances and Applications” http://www.research.att.com/~mgcr/doc/sgrasppr.pdf

Relinking Strategies

- *Periodical relinking* — not systematically applied to all solutions
- *Forward relinking* — worst solution is the initiating solution
- *Backward relinking* — best solution is the initiating solution
- *Backward and forward relinking* — both directions are explored
- *Mixed relinking* — relinking starts at both ends
- *Randomized relinking* — stochastic selection of moves
- *Truncated relinking* — the guiding solution is not reached

Related TS Methods

- Probabilistic Tabu Search
- Tabu Thresholding
- Reactive Tabu Search

Probabilistic Tabu Search

- Create move evaluations that include reference to tabu strategies, using penalties or inducements to modify a standard choice rule
- Map these evaluations to positive weights to obtain probabilities
- Chose the next move according to the probability values

Simple Tabu Thresholding

- Improving Phase
  - Construct *S**, the set of improving moves in the current neighborhood
  - If *S** is empty, execute the Mixed Phase. Otherwise select the probabilistic best move in *S**
- Mixed Phase
  - Select a value for the *TabuTiming* parameter
  - Select the probabilistic best move from the current neighborhood (full or reduced)
  - Continue for *TabuTiming* iterations and then return to Improving Phase

Some Tabu Thresholding Related Applications

- Bennell J. A. and K.A. Dowsland (1999) "A Tabu Thresholding Implementation for the Irregular Stock Cutting Problem," *International Journal of Production Research*, vol. 37, no. 18, pp. 4259-4275

- Kelly, J. P., M. Laguna and F. Glover (1994) "A Study of Diversification Strategies for the Quadratic Assignment Problem," *Computers and Operations Research*, vol. 21, no. 8, pp. 885-893.

- Valls, V., M. A. Perez and M. S. Quintanilla (1996) "Modified Tabu Thresholding Approach for the Generalized Restricted Vertex Coloring Problem," in Metaheuristics: Theory and Applications, I. H. Osman and J. P. Kelly (eds.), Kluwer Academic Publishers, pp. 537-554

- Vigo, D. and V. Maniezzo (1997) "A Genetic/Tabu Thresholding Hybrid Algorithm for the Process Allocation Problem," *Journal of Heuristics*, vol. 3, no. 2, pp. 91-110

Reactive Tabu Search

- Proposed by Battiti and Tecchiolli (1994)
- Based on keeping a record of all the solutions visited during the search
- Tabu tenure starts at 1 and is increased when repetitions are encountered and decreased when repetitions disappear
- Hashing and binary trees are used to identify repetitions

RTS Mechanisms

- Reaction Mechanism (Self-adjusting tabu tenure)
  - *CycleMax* (to trigger increases of the tabu tenure)
  - ← α (to calculate a moving average of the cycle length and control decreases of the tabu tenure)
  - *Increase* (a value greater than 1)
  - *Decrease* (a value less than 1)

- Escape Mechanism (Random moves)
  - *Rep* (repetition threshold)
  - *Chaos* (threshold to determine chaotic behavior)