Optimization. Tabu search

Maciej Komosinski

Institute of Computing Science
Poznan University of Technology
www.cs.put.poznan.pl/mkomosinski

The natural way of the creation of the algorithm

- Local search algorithm
  Inability to get out of local optima!

The natural way of the creation of the algorithm

- Local search algorithm
  Inability to get out of local optima!
- Acceptance of non-improving moves

The natural way of the creation of the algorithm

- Local search algorithm
  Inability to get out of local optima!
- Acceptance of non-improving moves
  Cycles!

The natural way of the creation of the algorithm

- Local search algorithm
  Inability to get out of local optima!
- Acceptance of non-improving moves
  Cycles!
- "Tabu" list

The natural way of the creation of the algorithm

- Local search algorithm
  Inability to get out of local optima!
- Acceptance of non-improving moves
  Cycles!
- "Tabu" list
  Too many banned moves!

The natural way of the creation of the algorithm

- Local search algorithm
  Inability to get out of local optima!
- Acceptance of non-improving moves
  Cycles!
- “Tabu” list
  Too many banned moves!
- Aspiration criteria – accepting banned but attractive moves

The natural way of the creation of the algorithm

- Local search algorithm
  Inability to get out of local optima!
- Acceptance of non-improving moves
  Cycles!
- "Tabu" list
  Too many banned moves!
- Aspiration criteria – accepting banned but attractive moves
  Tabu search algorithm

The natural way of the creation of the algorithm

- Local search algorithm
  Inability to get out of local optima!
- Acceptance of non-improving moves
  Cycles!
- "Tabu" list
  Too many banned moves!
- Aspiration criteria – accepting banned but attractive moves
  Tabu search algorithm
- Time-consuming evaluation of the entire neighborhood in each iteration?

The natural way of the creation of the algorithm

- Local search algorithm
  Inability to get out of local optima!
- Acceptance of non-improving moves
  Cycles!
- "Tabu" list
  Too many banned moves!
- Aspiration criteria – accepting banned but attractive moves
  Tabu search algorithm
- Time-consuming evaluation of the entire neighborhood in each iteration?
  "Candidates" – a subset of the neighborhood $V \subset N$

The natural way of the creation of the algorithm

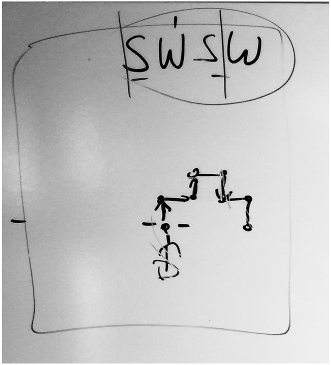
- Local search algorithm
  Inability to get out of local optima!
- Acceptance of non-improving moves
  Cycles!
- "Tabu" list
  Too many banned moves!
- Aspiration criteria – accepting banned but attractive moves
  Tabu search algorithm
- Time-consuming evaluation of the entire neighborhood in each iteration?
  "Candidates" – a subset of the neighborhood $V \subset N$

Tabu search

- the main idea – using memory

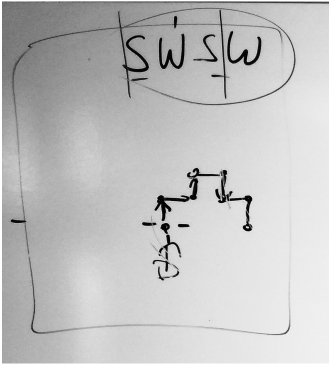
Tabu search

- the main idea – using memory



Tabu search

- the main idea – using memory



- remembering solutions or moves (changes)

An example of storing tabu moves (0)

Tabu list structure:

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

Inside: "tabu tenure" (the number of iterations until deactivation).

An example of storing tabu moves (0)

Tabu list structure:

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

Inside: "tabu tenure" (the number of iterations until deactivation).

Iteration 0 (starting point, maximization task)

2	5	7	3	4	6	1

Current solution –
value=10

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

move		Δ
5	4	6
7	4	4
3	6	2
2	3	0
4	1	-1

An example of storing tabu moves (1, 2)

Iteration 1

2	4	7	3	5	6	1

Current solution – value=16

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

move		Δ
3	1	2
2	3	1
3	6	-1
7	1	-2
6	1	-4

An example of storing tabu moves (1, 2)

Iteration 1

2	4	7	3	5	6	1

Current solution – value=16

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

move		Δ
3	1	2
2	3	1
3	6	−1
7	1	−2
6	1	−4

Iteration 2

2	4	7	1	5	6	3

Current solution – value=18

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

	move		Δ
T	1	3	−2
	2	4	−4
	7	6	−6
T	4	5	−7
	5	3	−9

An example of storing tabu moves (3, 4)

Iteration 3

4	2	7	1	5	6	3

Current solution – value=14

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

	move		Δ
T	4	5	6
	5	3	2
	7	1	0
	1	3	-3
	2	6	-6

An example of storing tabu moves (3, 4)

Super!

Iteration 3

4	2	7	1	5	6	3

Current solution – value=14

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

	move		Δ
T	4	5	6
	5	3	2
	7	1	0
	1	3	-3
	2	6	-6

An example of storing tabu moves (3, 4)

Super!

Iteration 3

4	2	7	1	5	6	3

Current solution – value=14

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

	move		Δ
T	4	5	6
	5	3	2
	7	1	0
	1	3	−3
	2	6	−6

Iteration 4

5	2	7	1	4	6	3

Current solution – value=20

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

move		Δ
7	1	0
4	3	−3
6	3	−5
5	4	−6
2	6	−8

Recency-based memory vs. frequency-based memory

The frequency of individual moves can be additionally used to disperse the search in the space of possible solutions (i.e., diversification). For example, moves can get a penalty proportional to their frequency if they don't improve the value of the solution.

Diversification is only useful under certain conditions (e.g., when there are no improvements).

Iteration 26. $\Delta' = \Delta - \text{frequency\_penalty}$

5	2	7	1	4	6	3

	1	2	3	4	5	6	7
1	·			3			
2		·					1
3	3		·				
4	2	5		·	2		
5		4		4	·		
6					1	·	
7	2			3			·

	move		Δ	Δ'
T	1	4	3	3
	2	4	-1	-6
	3	7	-3	-3
	1	6	-5	-5
	6	5	-6	-7

Tabu search

```
procedure TABU_SEARCH
begin
    INITIALIZE(xstart, xbest, T)
    x := xstart
    repeat
        GENERATE(V ⊂ N(x))
        SELECT(x')      //best f in V + aspiration
        UPDATE_TABU_LIST(T)
        if f(x') ≤ f(xbest) then xbest := x'
        x := x'
    until STOPPING_CONDITION
end
```

Determinism

The algorithm is deterministic.

- TS author: "a bad strategic choice is better than a good random choice" (because it is under control, so one can evaluate the strategy and draw conclusions)

New: a list of “candidates” – the V set

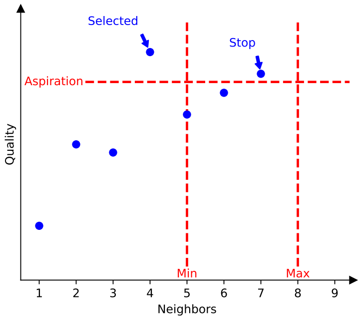
- what for: to avoid the need for generating and evaluating the entire neighborhood in each iteration
- a good move, if not applied in the current iteration, will still be good in the next few iterations (?)

New: a list of "candidates" – the V set

- what for: to avoid the need for generating and evaluating the entire neighborhood in each iteration
- a good move, if not applied in the current iteration, will still be good in the next few iterations (?)
- which subset $V \subset N$ of the set of neighbors N should constitute the subset of candidates?
  - candidates = good neighbors
  - we need to choose the moves that are beneficial... for the current solution and for future ones.

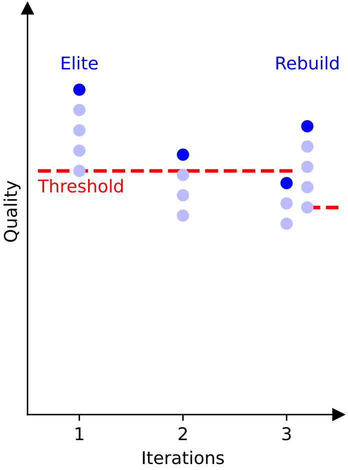
Construction of the list of candidates. The idea of strategy 1: "Aspiration plus"

- searching the neighborhood until a neighbor is found better by a certain threshold value ("aspiration plus")
- the number of candidates increases until the threshold value is reached
- Min $\leq$ number of visited neighbors $\leq$ Max
- aspiration level may vary during the search (may depend on the search history)
- the strategy returns 1 or more best neighbors found
- as many as 3 parameters...
- details: [?]



Construction of the list of candidates. The idea of strategy 2: "Elite candidate list"

- to build the list, check all or most of the moves and select the best k of them (k is a parameter)
- in subsequent iterations, the currently best move from the list is applied until the quality of the move drops below a given threshold, or a certain number of iterations is reached
- can be adaptive
- details: [**?**]



Aspiration criteria

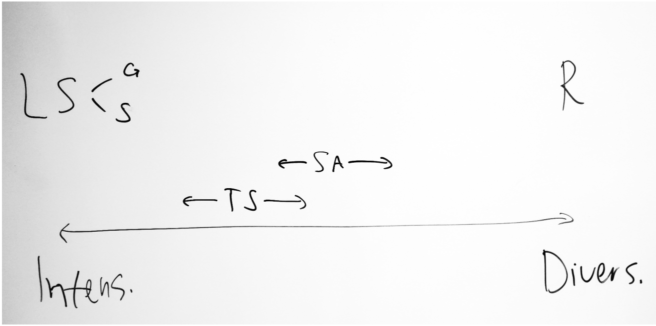
- goal: to decide when tabu restrictions can be overridden
- basic aspiration criterion (by optimization objective, global, shown in the example in the beginning of this presentation): remove the tabu constraint when the move yields a solution better than the best solution found so far
- **aspiration by default**
  - if all moves are tabu and they are not allowed by other criteria, then the move that is the least tabu is selected
- **aspiration by optimization objective**
  - global – the move $x \rightarrow x'$ that is tabu is accepted if $cost(x') < best\_cost$
  - regional (the solution space is divided into regions R) – the move that is tabu is accepted if $cost(x') < best\_cost(R)$. R is the region where x' is located.

Unification of optimization algorithms

Intensification and diversification (exploitation and exploration)

Unification of optimization algorithms

Intensification and diversification (exploitation and exploration)



- Discussion: is higher intensification than LS possible?
- EA? RW? RS with forced diversification?
- Can both properties be simultaneously improved? 2D?
- What happens to exploitation and exploration when "the fundamental premise of optimization" fades away?

Intensification and diversification in *Tabu Search*

Intensification and diversification in *Tabu Search*

- intensification (exploitation)
  - in areas with good solutions
  - coming back to the best solution found so far
  - short term memory – shortening the tabu list
  - long term memory
    - each solution or move is a collection of components
    - remembering the components of good moves or solutions during optimization
    - during the intensification period, moves or solutions incorporate the good components
    - long-term memory enables "learning"
- diversification (exploration)
  - for rarely visited areas
  - penalizing frequent moves – escaping from the area
- these mechanisms can be perceived as a way of modifying the objective function: $f' = f + Int + Div$

References I

Fred Glover, Manuel Laguna, Panos Pardalos, D.-Z. Du, and R. L. Graham.

Tabu search: effective strategies for hard problems in analytics and computational science.

Handbook of Combinatorial Optimization, 2nd ed, vol. XXI, pages 3261–3362, 2013.

doi:10.1007/978-1-4419-7997-1_17.