Problem Solving and Search in Artificial Intelligence

Tabu Search

Nysret Musliu
Database and Artificial Intelligence Group,
Institut für Informationssysteme, TU-Wien
Introduction

- Local search techniques
  - Tabu search
  - Simulated annealing
  - Stochastic Hill-Climber
  - ...
- Tabu Search uses memory during the search
- In memory are stored relevant information about the history of search
- The memory should help to avoid the cycles during the search
- Tabu search is a deterministic heuristic technique
Basic Tabu Search

Procedure Tabu-Suche
begin
  Initialize tabu list
Basic Tabu Search

Procedure Tabu-Suche
begin
  Initialize tabu list
  Generate randomly Initial Solution $s_c$
Basic Tabu Search

Procedure Tabu-Suche
begin
  Initialize tabu list
  Generate randomly Initial Solution $s_c$
  Evaluate $s_c$
Basic Tabu Search

Procedure Tabu-Suche
begin
  Initialize tabu list
  Generate randomly Initial Solution $s_c$
  Evaluate $s_c$
repeat
  Generate all neighborhood solutions of the solution $s_c$
Basic Tabu Search

Procedure Tabu-Suche
begin
    Initialize tabu list
    Generate randomly Initial Solution $s_c$
    Evaluate $s_c$
repeat
    Generate all neighborhood solutions of the solution $s_c$
    Find best solution $s_x$ in the neighborhood
Basic Tabu Search

Procedure Tabu-Suche
begin
  Initialize tabu list
  Generate randomly Initial Solution \( s_c \)
  Evaluate \( s_c \)
repeat
  Generate all neighborhood solutions of the solution \( s_c \)
  Find best solution \( s_x \) in the neighborhood
  if \( s_x \) is not tabu solution then \( s_c = s_x \)
Basic Tabu Search

Procedure Tabu-Suche
begin
  Initialize tabu list
  Generate randomly Initial Solution $s_c$
  Evaluate $s_c$
repeat
  Generate all neighborhood solutions of the solution $s_c$
  Find best solution $s_x$ in the neighborhood
  if $s_x$ is not tabu solution then $s_c = s_x$
  else if 'aspiration criteria' is fulfilled then
    $s_c = s_x$
Basic Tabu Search

Procedure Tabu-Suche
begin
  Initialize tabu list
  Generate randomly Initial Solution $s_c$
  Evaluate $s_c$
repeat
  Generate all neighborhood solutions of the solution $s_c$
  Find best solution $s_x$ in the neighborhood
  if $s_x$ is not tabu solution then $s_c = s_x$
  else if 'aspiration criteria' is fulfilled then
    $s_c = s_x$
  else
    find best not tabu solution in the neighborhood $s_{nt}$
    $s_c = s_{nt}$
Basic Tabu Search

Procedure Tabu-Suche
begin
    Initialize tabu list
    Generate randomly Initial Solution $s_c$
    Evaluate $s_c$
repeat
    Generate all neighborhood solutions of the solution $s_c$
    Find best solution $s_x$ in the neighborhood
    if $s_x$ is not tabu solution then $s_c = s_x$
    else if 'aspiration criteria' is fulfilled then
        $s_c = s_x$
    else
        find best not tabu solution in the neighborhood $s_{nt}$
        $s_c = s_{nt}$
    Update tabu list
until (terminate-condition)
end
Example

- SAT problem:
  - Make a compound statement of Boolean variables to evaluate to true
  - Suppose we have to solve the SAT problem with 8 variables:
    \[ F(x) = (x_1 \lor \overline{x}_3 \lor x_7) \land (\overline{x}_1 \lor x_2) \land \ldots \land (\overline{x}_2 \lor x_4 \lor x_7) \]
  - Find the truth assignment for each variable \( x_i \) such that \( F(x) = TRUE \)
General questions

- Representation of solution
  - Candidate solution is represented with a binary string of length $n$ (number of variables)
    - Example: $X=(0,0,0,1,1,1,0,1)$ represents this solution: $x_1=0$, $x_2=0$, $x_3=0$, $x_4=1$, $x_5=1$, $x_6=1$, $x_7=0$, $x_8=1$

- Evaluation of function:
  - Weighted sum of a number of satisfied clauses (weights depend on the number of variables in clause)

- Initial Solution
  - Can be generated for example by random assignment of variables with 0 or 1: $X=(0,1,1,1,0,0,0,0)$
Neighborhood generation

- **Moves**
  - A simple move is defined, which flips the value of one variable from 1 to 0 or from 0 to 1
  - More moves can be defined…
- If we apply only the first move the whole neighborhood of solution can be generated by flipping of value of each variable
- In tabu search usually the whole neighborhood is generated during each iteration
Tabu search specific questions

- Memory
  - Which information should we store during the search to possibly avoid the cycles?
Memory

- Recency-based memory
  - Some parameters of few past iterations are stored
    - For example for SAT problem we could store the information for the flipped variables in past 5 iterations
  - Based on that we could forbid (make tabu) the flipping of variables which were flipped in last 5 iterations

Variable $x_3$ should not be flipped in next 2 iterations

Variable $x_7$ should not be flipped in next 4 iterations

00200040
Memory

- Frequency-based memory
  - Stores information for larger number of iterations
    - For example for SAT problem we could store the information about number of flips for each variable during the last 100 iterations
  - Based on that we could prefer some of flips of variables more than others during the search

Variable x2 has been flipped 3 times in past 100 iterations

Variable x5 has been flipped 12 times in past 100 iterations

12 3 35 20 10 15 12 3
Selection of solutions

- The acceptance of solution for next iteration depends not only from its quality
- The memory has also the impact in the selection process
- Solution are classified in tabu and not tabu solutions
- Usually the best non tabu solutions is accepted for the next iteration
Selection of solutions

- Aspiration criteria
  - Tabu solution may be accepted if it fulfills some conditions
    - Example: The tabu solution is the best solution so far

- Based on frequency based memory
  - Search can be intensified
    - More frequent moves are preferred
  - Search can be diversified
    - Less used moves during the search are preferred
SAT example

Procedure Tabu-Suche
begin

*Initialize tabu list*
Generate randomly Initial Solution $s_c$
Evaluate $s_c$
repeat
  Generate all neighborhood solutions of the solution $s_c$
  Find best solution $s_x$ in the neighborhood
  if $s_x$ is not tabu solution then $s_c = s_x$
  else if 'aspiration criteria' is fulfilled then
    $s_c = s_x$
  else
    find best not tabu solution in the neighborhood $s_{nt}$
    $s_c = s_{nt}$
  Update tabu list
until (terminate-condition)
end
SAT Problem

- Initialize memory
  - Recency based memory
    - M:
      - M: 
      - F:  
  - Frequency based memory
    - F: 
      - F:  

SAT example

Procedure Tabu-Suche
begin
    Initialize tabu list
    **Generate randomly Initial Solution** $s_c$
    Evaluate $s_c$
repeat
    Generate all neighborhood solutions of the solution $s_c$
    Find best solution $s_x$ in the neighborhood
    if $s_x$ is not tabu solution then $s_c = s_x$
    else if 'aspiration criteria' is fulfilled then
        $s_c = s_x$
    else
        find best not tabu solution in the neighborhood $s_{nt}$
        $s_c = s_{nt}$
    Update tabu list
until (terminate-condition)
end
SAT example

- Initial Solution
  - Random generated solution

\[ s_c : \begin{array}{cccccccc}
1 & 0 & 1 & 0 & 0 & 0 & 1 & 1
\end{array} \]
SAT example

Procedure Tabu-Suche
begin
   Initialize tabu list
   Generate randomly Initial Solution $s_c$

   Evaluate $s_c$
repeat
   Generate all neighborhood solutions of the solution $s_c$
   Find best solution $s_x$ in the neighborhood
   if $s_x$ is not tabu solution then $s_c = s_x$
   else if 'aspiration criteria' is fulfilled then
      $s_c = s_x$
   else
      find best not tabu solution in the neighborhood $s_{nt}$
      $s_c = s_{nt}$
   Update tabu list
until (terminate-condition)
end
SAT example

- Evaluate solution
  - Suppose that the fitness of solution is 30
SAT example

Procedure Tabu-Suche
begin
    Initialize tabu list
    Generate randomly Initial Solution \( s_c \)
    Evaluate \( s_c \)
repeat
    Generate all neighborhood solutions of the solution \( s_c \)
    Find best solution \( s_x \) in the neighborhood
    if \( s_x \) is not tabu solution then \( s_c = s_x \)
    else if 'aspiration criteria' is fulfilled then
        \( s_c = s_x \)
    else
        find best not tabu solution in the neighborhood \( s_{nt} \)
        \( s_c = s_{nt} \)
    Update tabu list
until (terminate-condition)
end
Neighborhood of current solution

\[ s_c = \begin{array}{ccccccc}
1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\
\end{array} \]

\[ s_1 = \begin{array}{ccccccc}
0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\
\end{array} \]
Neighborhood of current solution

<table>
<thead>
<tr>
<th></th>
<th>s_c</th>
<th>s_1</th>
<th>s_2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1 1 0 1 0 0 0 1 1</td>
<td>0 0 1 0 0 0 1 1</td>
<td>1 1 1 0 0 0 1 1</td>
</tr>
</tbody>
</table>
Neighborhood of current solution

Eight neighborhood solutions, which are obtained by flipping of a single bit in the solution $s_c$.

<table>
<thead>
<tr>
<th>$s_c$</th>
<th>1 0 1 0 0 0 1 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s_1$</td>
<td>0 0 1 0 0 0 1 1</td>
</tr>
<tr>
<td>$s_2$</td>
<td>1 1 1 0 0 0 1 1</td>
</tr>
<tr>
<td>$s_3$</td>
<td>1 0 0 0 0 0 1 1</td>
</tr>
<tr>
<td>$s_4$</td>
<td>1 0 1 1 0 0 1 1</td>
</tr>
<tr>
<td>$s_5$</td>
<td>1 0 1 0 1 0 1 1</td>
</tr>
<tr>
<td>$s_6$</td>
<td>1 0 1 0 0 1 1 1</td>
</tr>
<tr>
<td>$s_7$</td>
<td>1 0 1 0 0 0 0 1</td>
</tr>
<tr>
<td>$s_8$</td>
<td>1 0 1 0 0 0 1 0</td>
</tr>
</tbody>
</table>
SAT example

Procedure Tabu-Suche
begin
  Initialize tabu list
  Generate randomly Initial Solution $s_c$
  Evaluate $s_c$
repeat
  Generate all neighborhood solutions of the solution $s_c$
  Find best solution $s_x$ in the neighborhood
  if $s_x$ is not tabu solution then $s_c = s_x$
  else if 'aspiration criteria' is fulfilled then
    $s_c = s_x$
  else
    find best not tabu solution in the neighborhood $s_{nt}$
    $s_c = s_{nt}$
  Update tabu list
until (terminate-condition)
end
Evaluation of solutions

<table>
<thead>
<tr>
<th>$s_c$</th>
<th>1</th>
<th>0</th>
<th>1</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>1</th>
<th>1</th>
<th>Evaluation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s_1$</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>31</td>
</tr>
<tr>
<td>$s_2$</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>37</td>
</tr>
<tr>
<td>$s_3$</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>34</td>
</tr>
<tr>
<td>$s_4$</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>29</td>
</tr>
<tr>
<td>$s_5$</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>32</td>
</tr>
<tr>
<td>$s_6$</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>28</td>
</tr>
<tr>
<td>$s_7$</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>29</td>
</tr>
<tr>
<td>$s_8$</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>33</td>
</tr>
</tbody>
</table>
## Best solution

<table>
<thead>
<tr>
<th>$s_c$</th>
<th>1</th>
<th>0</th>
<th>1</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>1</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s_1$</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$s_2$</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$s_3$</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$s_4$</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$s_5$</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$s_6$</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$s_7$</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$s_8$</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

### Evaluation
- $s_c$: 29
- $s_2$: 37
- $s_1$: 31
- $s_3$: 34
- $s_4$: 29
- $s_5$: 32
- $s_6$: 28
- $s_7$: 29
- $s_8$: 33

*Best solution*
SAT example

Procedure Tabu-Suche
begin
Initialize tabu list
Generate randomly Initial Solution $s_c$
Evaluate $s_c$
repeat
Generate all neighborhood solutions of the solution $s_c$
Find best solution $s_x$ in the neighborhood
if $s_x$ is not tabu solution then $s_c = s_x$
else if 'aspiration criteria' is fulfilled then
    $s_c = s_x$
else
    find best not tabu solution in the neighborhood $s_{nt}$
    $s_c = s_{nt}$

Update tabu list
until (terminate-condition)
end
Update memory

- Recency based memory

- $M: \begin{bmatrix} 0 & 5 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$
Update memory

- Recency based memory

- M: 0 5 0 0 0 0 0 0

Flipping of bit in position 2 is Tabu in next 5 iterations
Update memory

- Recency based memory
  - M: \[0 \ 5 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0\]

- Frequency based memory
  - F: \[0 \ 1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0\]

Flipping of bit in position 2 is Tabu in next 5 iterations
Update memory

- Recency based memory
  - M: 
    - 0 5 0 0 0 0 0 0

- Frequency based memory
  - F: 
    - 0 1 0 0 0 0 0 0

Flipping of bit in position 2 is Tabu in next 5 iterations

Bit in position 2 has been flipped one time
Update memory

- Suppose that in next iteration the best solution is obtained by flipping the bit in position 4. The content of memory after the second iteration will be:

\[
\begin{align*}
M: & \quad 0 4 0 5 0 0 0 0 0 \\
F: & \quad 0 1 0 1 0 0 0 0 0
\end{align*}
\]

All non zeros entries are decreased by one at every iteration
SAT problem

- Suppose that after 8 iterations the short term memory has following content:

\[ M: \begin{array}{cccccccc}
0 & 4 & 5 & 3 & 1 & 0 & 2 & 0 \\
\end{array} \]

- Suppose that the following solutions are obtained from the neighborhood of the current solution:
  - \( \text{eval}(s_1)=35, \text{eval}(s_2)=38, \text{eval}(s_3)=36, \text{eval}(s_4)=34, \text{eval}(s_5)=32, \text{eval}(s_6)=30, \text{eval}(s_7)=34, \text{eval}(s_8)=33 \)
SAT problem

- Suppose that after 8 iterations the short term memory has following content:

  M: \[0 \ 4 \ 5 \ 3 \ 1 \ 0 \ 2 \ 0\]

- The following solutions are obtained from the neighborhood of the current solution:
  - \( \text{eval}(s_1) = 35 \), \( \text{eval}(s_2) = 38 \), \( \text{eval}(s_3) = 36 \), \( \text{eval}(s_4) = 34 \),
  - \( \text{eval}(s_5) = 32 \), \( \text{eval}(s_6) = 30 \), \( \text{eval}(s_7) = 34 \), \( \text{eval}(s_8) = 33 \)

- Best solution in neighborhood has the fitness 38, but it is obtained by flipping bit 2

Flip of bit 2 is tabu! Should we accept this solution?
SAT example

Procedure Tabu-Suche
begin
  Initialize tabu list
  Generate randomly Initial Solution $s_c$
  Evaluate $s_c$
repeat
  Generate all neighborhood solutions of the solution $s_c$
  Find best solution $s_x$ in the neighborhood
  if $s_x$ is not tabu solution then $s_c = s_x$
    else if 'aspiration criteria' is fulfilled then
      $s_c = s_x$
      else
        find best not tabu solution in the neighborhood $s_{nt}$
        $s_c = s_{nt}$
  Update tabu list
until (terminate-condition)
end
Aspiration criteria

- The tabu solution may be accepted, if it fulfills some conditions
  - For example if the solution is the best solution found so far
- Suppose that in SAT Example the best solution found is far has fitness 39
Procedure Tabu-Suche
begin
    Initialize tabu list
    Generate randomly Initial Solution $s_c$
    Evaluate $s_c$
repeat
    Generate all neighborhood solutions of the solution $s_c$
    Find best solution $s_x$ in the neighborhood
    if $s_x$ is not tabu solution then $s_c = s_x$
    else if 'aspiration criteria' is fulfilled then
    $s_c = s_x$
    else
        find best not tabu solution in the neighborhood $s_{nt}$
        $s_c = s_{nt}$
    Update tabu list
until (terminate-condition)
end
SAT example

M: 0 4 5 3 1 0 2 0

- eval(s₁)=35, eval(s₂)=38, eval(s₃)=36, eval(s₄)=34, eval(s₅)=32, eval(s₆)=30, eval(s₇)=34, eval(s₈)=33

- If the aspiration criteria is not fulfilled, only non tabu solutions will be taken in consideration
  - Solutions: s₁, s₆, s₈
SAT example

M: 0 4 5 3 1 0 2 0

- eval(s₁)=35, eval(s₂)=38, eval(s₃)=36, eval(s₄)=34, eval(s₅)=32, eval(s₆)=30, eval(s₇)=34, eval(s₈)=33

- If the aspiration criteria is not fulfilled, only non tabu solutions will be taken in consideration
  - Solutions: s₁, s₆, s₈

Solution s₁ is accepted for the next iteration
SAT example

Procedure Tabu-Suche
begin
  Initialize tabu list
  Generate randomly Initial Solution $s_c$
  Evaluate $s_c$
repeat
  Generate all neighborhood solutions of the solution $s_c$
  Find best solution $s_x$ in the neighborhood
  if $s_x$ is not tabu solution then $s_c = s_x$
  else if 'aspiration criteria' is fulfilled then
    $s_c = s_x$
  else
    find best not tabu solution in the neighborhood $s_{nt}$
    $s_c = s_{nt}$
  Update tabu list
until (terminate-condition)
end
Termination condition

- Optimal solution found
- Number of iterations
- Time
- Empty Neighborhood
- No improves of solution for a determined time/number of iterations
- User interaction
- …
Use of frequency-based memory

Suppose that the content of frequency-based memory for SAT problem after 100 iterations has the following content:

$$F: \begin{array}{cccccccc}
12 & 11 & 15 & 10 & 11 & 11 & 27 & 3 \\
\end{array}$$

The following solutions are obtained from the neighborhood of the current solution:

- $eval(s_1)=46$, $eval(s_2)=43$, $eval(s_3)=46$, $eval(s_4)=45$, $eval(s_5)=44$, $eval(s_6)=43$, $eval(s_7)=46$, $eval(s_8)=46$
Use of frequency-based memory

- Suppose that the content of frequency-based memory for SAT problem after 100 iterations has the following content:
  
  | F: 12 | 11 | 15 | 10 | 11 | 11 | 27 | 3 |
  
- The following solutions are obtained from the neighborhood of the current solution:
  - eval(s₁)=46, eval(s₂)=43, eval(s₃)=46, eval(s₄)=45, eval(s₅)=44, eval(s₆)=43, eval(s₇)=46, eval(s₈)=46

- Suppose that only solutions s₃, s₇, s₈ are non-tabu solutions
Use of frequency-based memory

\[ F: \begin{array}{cccccccc}
12 & 11 & 15 & 10 & 11 & 11 & 27 & 3 \\
\end{array} \]

eval(s_1)=46, eval(s_2)=43, eval(s_3)=46, eval(s_4)=45, eval(s_5)=44, eval(s_6)=43, eval(s_7)=46, eval(s_8)=46

- Solutions s_3, s_7, s_8 are non-tabu solutions
- Possible use of memory
  - Make less frequently used moves more attractive
  - Diversification of search
Use of frequency-based memory

F: 12 11 15 10 11 11 27 3

eval(s_1)=46, eval(s_2)=43, eval(s_3)=46, eval(s_4)=45, eval(s_5)=44, eval(s_6)=43, eval(s_7)=46, eval(s_8)=46

- Solutions s_3, s_7, s_8 are non-tabu solutions
- Possible use of memory
  - Make less frequently used moves more attractive
    - Diversification of search

Solution s_8 will be accepted for next iteration
Use of frequency-based memory

- Other possibilities of use of frequency-based memory
  - Aspiration by default
    - Select a move that is the “oldest” of all considered
  - Aspiration by search direction
    - Memorize also whether or not the moves generated improvements
  - Aspiration by influence
    - Particular move can have larger influence if a “larger” step is made from the old solution to the new
Tabu list

- Length of tabu list (for how many iteration should the solution be made tabu)
  - Usually depends from size of problem
  - The length of tabu list could also change during the search
    - Reactive tabu search
Tabu List

- Hashing
- FIFO list
- Storage of last usage time of moves
Adaptive length of tabu list

- Length is 1 in the beginning
- Length increases when the repetitions of solutions happens
- Length decreases when the repetition of solutions disappears
Literature


