Improved methods for the Travelling Salesperson with Hotel Selection

M. Castro \(^1\)  K. Sörensen \(^1\)  P. Vansteenwegen \(^2\)  P. Goos \(^1\)

\(^1\)ANT/OR, University of Antwerp, Belgium

\(^2\)Department of Industrial Management, Ghent University, Belgium

ORBEL 26 - February 2nd, 2012
Contents

1 Problem description
2 Solution technique
3 Computational experiments
4 Conclusions
Contents

1 Problem description

2 Solution technique

3 Computational experiments

4 Conclusions
Problem description

- A new variant of the TSP called Traveling Salesman Problem with Hotel Selection (TSP-HS)
- A TSP tour is cut into trips no longer than a day, each of them staring and ending at a hotel
- A tour is a set of connected trips that combined visit all customers
- The objective is to minimize the number of connected trips and then minimize the total length of the tour
An instance with 30 customers - A TSP tour
An instance with 30 customers and 4 hotels
A TSP-HS tour containing three trips
The TSP-HS has a number of interesting applications:

- A travelling salesperson
- Truck drivers
- Planning touristic multi-day tours
- Google Street View
Mathematical formulation

- Given a set of $s + 1$ hotels ($i = 0, \ldots, s$) and $n$ customers ($i = s + 1, \ldots, s + n$), let $c_{i,j}$ be the time needed to travel from location $i$ to location $j$.
- Let $x_{i,j,d}$ be a binary variable that takes the value of 1 if salesperson visit point $j$ immediately after point $i$ on trip $d$, or 0 if not.
- Let $y_d$ be a binary variable that denotes if the trip $d$ is needed or not.
- Each customer is assigned a service time $T_i$.
- The available time that each trip takes is limited to a given time budget $C$. 
Mathematical formulation

- Let \( N \) be the set of customers
- Let \( D \) denote the maximum number of trips and let \( M \) be a big number
- Each trips should start and end in one of the available hotels
- The ending hotel on day \( i \) has to be equal to the starting hotel on day \( i + 1 \)
- The starting and ending hotel of the tour are assumed to be identical and given \((i = 0)\)
Mathematical formulation

minimize \[ M \sum_{d=1}^{D} y_d + \sum_{i=0}^{s+n} \sum_{j=0}^{s+n} \sum_{d=1}^{D} c_{i,j} x_{i,j,d} \]

subject to \[ \sum_{d=1}^{D} x_{i,j,d} = 1, \text{ for } j = s + 1, \ldots, s + n \]
\[ \sum_{d=1}^{D} x_{i,j,d} = \sum_{i=0}^{s+n} x_{j,i,d}, \text{ for } j = s + 1, \ldots, s + n, \text{ } d = 1, \ldots, D \]
\[ \sum_{i=0}^{s+n} \sum_{j=0}^{s+n} (c_{i,j} + T_j) x_{i,j,d} \leq C, \text{ for } d = 1, \ldots, D \]
Mathematical formulation

\[\sum_{h=0}^{s} \sum_{j=0}^{s+n} x_{h,j,d} = \sum_{h=0}^{s} \sum_{i=0}^{s+n} x_{i,h,d} = y_d, \text{ for } d = 1, \ldots, D\]

\[\sum_{j=1}^{s+n} x_{0,j,1} = 1\]

\[\sum_{i=1}^{s+n} x_{i,0,d} \geq y_d - y_{d+1}, \text{ for } d = 1, \ldots, D - 1\]

\[x_{i,j,d} \leq y_d, \text{ for } i,j = 0, \ldots, s+n, d = 1, \ldots, D\]

\[y_d \geq y_{d+1}, \text{ for } d = 1, \ldots, D - 1\]
Mathematical formulation

\[ \begin{align*}
\sum_{i=0}^{s+n} x_{i,h,d} + y_d & \geq \sum_{j=0}^{s+n} x_{h,j,d+1} + y_{d+1}, \text{ for } d = 1, \ldots, D - 1, h = 1, \ldots, s \\
\sum_{i=0}^{s+n} x_{i,h,d} - \sum_{j=0}^{s+n} x_{h,j,d+1} & \leq 1 - y_{d+1}, \text{ for } h = 0, \ldots, s, d = 1, \ldots, D - 1 \\
\sum_{j \in S} \sum_{j \in S} x_{i,j,d} & \leq |S| - 1 \text{ for } S \subset N, 2 \leq |S| \leq n - 1 \\
x_{i,j,d} & \in \{0, 1\}, \quad y_d \in \{0, 1\}
\end{align*} \]
Contents

1 Problem description

2 Solution technique

3 Computational experiments

4 Conclusions
How to solve it?

- Exact algorithms (e.g. B&B, B&C, ...)
  - Able to solve small instances ($\leq 40$)
  - Impractical!

- Metaheuristics
  - Good solutions in very limited computing times
Solution technique

- A multi-start procedure
  - Variable Neighbourhood Descend (VND)
- The algorithm implements well-known heuristics for solving optimization problems like TSP and VRP
  - involving a single trip, e.g. 2-opt
  - involving two trips, e.g. Relocation
The main procedure performs as follows:

1: `repeat`
2: Generate an initial solution
3: Improve initial solution
4: `until` Maximum number of iterations is reached
5: `return` Best solution found
Pseudo-code for the generation of initial solutions

1: repeat
2: Generate a complete TSP tour in a greedy/randomized way
3: Improve tour
4: Split the TSP tour into trips no longer than the maximum allowed length \( (C) \)
5: until Maximum number of solutions has reached
6: return Best initial solution found
Heuristics for improving a trip

Tested with different operators/heuristics:
- Reinsertions
- Edges exchange
- Lin-Kernighan heuristic
Operators for improving tours

Classical moves (VRP)
- Relocate
- Exchange

Specific moves
- Change hotels
- Join Trips
Change hotels

- Look at every pair of consecutive trips and it tries to change the intermediate hotel
Join trips

- Attempts to reduce the number of trips by joining two trips sharing the same hotel
Improving solutions

Once a “good” initial solution is generated, then it is improved:

1. repeat
2. Relocate
3. Exchange
4. Change hotels
5. Join trips
6. until No improvement is possible
7. Update best solution
Contents

1 Problem description

2 Solution technique

3 Computational experiments

4 Conclusions
Computational experiments

- Set 1: made up of benchmarks for CVRP and MDVRP
  - Between 48 and 288 customers
- Set 2
  - Small instances
- Set 3: made up of benchmarks for the TSP
  - Between 51 and 1002 customers
- Set 4
  - Same instances as in Set 3, with arbitrary budget
The heuristic is controlled by 5 parameters:

- $S$ Number of iterations
- $I$ Number of initial solutions
- $c$ Size of the candidate list
- $Opt1$ Operator used in the construction phase
- $Opt2$ Operator used in the improvement phase
Parametric analysis

The possible values for $Opt_1$ and $Opt_2$ are categorised as follows:

1. 3-opt
2. 2-opt
3. 2-opt/Or-opt
4. 2-opt/Re-opt
An experiment was conducted with the aim to determine the most robust configuration of parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Values</th>
<th>Num. of levels</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S$</td>
<td>5, 10, 20, 30</td>
<td>4</td>
</tr>
<tr>
<td>$I$</td>
<td>5, 25, 50, 100</td>
<td>4</td>
</tr>
<tr>
<td>$c$</td>
<td>3, 5</td>
<td>2</td>
</tr>
<tr>
<td>$Opt_1$</td>
<td>2, 3, 4</td>
<td>3</td>
</tr>
<tr>
<td>$Opt_2$</td>
<td>1, 2, 3, 4</td>
<td>4</td>
</tr>
</tbody>
</table>

Table: Parameters
A full-factorial experiment was conducted on each of 30 different instances picked from sets 1, 3 and 4
- 11520 observations ($= 4^3 \times 3 \times 2 \times 30$)
- The results were analysed by performing a multi-way analysis of variance (ANOVA)
### Parametric analysis

<table>
<thead>
<tr>
<th>Source</th>
<th>Nparm</th>
<th>DF</th>
<th>DFDen</th>
<th>F Ratio</th>
<th>Prob &gt; F</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>3</td>
<td>3</td>
<td>11422</td>
<td>115.0732</td>
<td>&lt; 0.0001</td>
</tr>
<tr>
<td>I</td>
<td>3</td>
<td>3</td>
<td>11422</td>
<td>73.0369</td>
<td>&lt; 0.0001</td>
</tr>
<tr>
<td>c</td>
<td>1</td>
<td>1</td>
<td>11422</td>
<td>12.7682</td>
<td>0.0004</td>
</tr>
<tr>
<td>Opt1</td>
<td>2</td>
<td>2</td>
<td>11422</td>
<td>37.6057</td>
<td>&lt; 0.0001</td>
</tr>
<tr>
<td>Opt2</td>
<td>3</td>
<td>3</td>
<td>11422</td>
<td>0.8670</td>
<td>0.4573</td>
</tr>
<tr>
<td>S*I</td>
<td>9</td>
<td>9</td>
<td>11422</td>
<td>4.2754</td>
<td>&lt; 0.0001</td>
</tr>
<tr>
<td>S*c</td>
<td>3</td>
<td>3</td>
<td>11422</td>
<td>0.6540</td>
<td>0.5803</td>
</tr>
<tr>
<td>S*Opt1</td>
<td>6</td>
<td>6</td>
<td>11422</td>
<td>1.0860</td>
<td>0.3680</td>
</tr>
<tr>
<td>S*Opt2</td>
<td>9</td>
<td>9</td>
<td>11422</td>
<td>0.8604</td>
<td>0.5602</td>
</tr>
<tr>
<td>I*c</td>
<td>3</td>
<td>3</td>
<td>11422</td>
<td>0.7973</td>
<td>0.4951</td>
</tr>
<tr>
<td>I*Opt1</td>
<td>6</td>
<td>6</td>
<td>11422</td>
<td>4.9165</td>
<td>&lt; 0.0001</td>
</tr>
<tr>
<td>I*Opt2</td>
<td>9</td>
<td>9</td>
<td>11422</td>
<td>0.4649</td>
<td>0.8989</td>
</tr>
<tr>
<td>c*Opt1</td>
<td>2</td>
<td>2</td>
<td>11422</td>
<td>0.7376</td>
<td>0.4783</td>
</tr>
<tr>
<td>c*Opt2</td>
<td>3</td>
<td>3</td>
<td>11422</td>
<td>0.1762</td>
<td>0.9126</td>
</tr>
<tr>
<td>Opt1*Opt2</td>
<td>6</td>
<td>6</td>
<td>11422</td>
<td>0.3195</td>
<td>0.9271</td>
</tr>
</tbody>
</table>
Mean plot for the number of initial solutions
Mean plot for Opt1

![Graph showing trips and time for Opt1]

- Trips: 7.2, 7.5, 7.6
- Time: 0.2, 0.4, 0.6
## Parameter settings

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S$</td>
<td>5</td>
</tr>
<tr>
<td>$I$</td>
<td>50</td>
</tr>
<tr>
<td>$c$</td>
<td>3</td>
</tr>
<tr>
<td>$Opt1$</td>
<td>2</td>
</tr>
<tr>
<td>$Opt2$</td>
<td>1</td>
</tr>
</tbody>
</table>

Table: Settings
Results for Set 1

Summary

- Instances: 16
- Customers: between 48 and 288
- Hotels: 6
- Optimal solutions not known
- New best known: 16
- Gap:
  - Max: 4.80 %
  - Min: 0.34 %
  - Avg: 1.79 %
Results for Set 2

Summary

- Instances: 52
  - 4 groups of 13 instances for $N \in \{10, 15, 30, 40\}$ customers
- Hotels: 2
- Solutions
  - 47 optimal solutions
Results for Set 2

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>13</td>
<td>13</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>5</td>
<td>0.03</td>
<td>7.28</td>
<td>3.30</td>
</tr>
<tr>
<td>15</td>
<td>13</td>
<td>13</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>5</td>
<td>0.25</td>
<td>6.31</td>
<td>2.05</td>
</tr>
<tr>
<td>30</td>
<td>11</td>
<td>6</td>
<td>0.08</td>
<td>2.92</td>
<td>0.88</td>
<td>1</td>
<td>0.48</td>
<td>5.78</td>
<td>3.25</td>
</tr>
<tr>
<td>40</td>
<td>10</td>
<td>5</td>
<td>0.16</td>
<td>2.36</td>
<td>1.10</td>
<td>0</td>
<td>0.01</td>
<td>6.85</td>
<td>3.70</td>
</tr>
</tbody>
</table>
Results for Set 3

Summary

- Instances: 48
  - 3 groups of 16 instances for $M = 1 + s, (s \in \{3, 5, 10\})$ hotels
- Customers: between 51 and 1002
- Optimal solutions available
Results for Set 3

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>2</td>
<td>0.30</td>
<td>8.26</td>
<td>4.72</td>
<td>0</td>
<td>2.87</td>
<td>17.38</td>
<td>12.73</td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td>0.66</td>
<td>9.62</td>
<td>5.40</td>
<td>0</td>
<td>6.40</td>
<td>22.37</td>
<td>13.82</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>0.54</td>
<td>8.88</td>
<td>5.84</td>
<td>0</td>
<td>9.85</td>
<td>36.49</td>
<td>17.93</td>
</tr>
</tbody>
</table>
Results for Set 4

Summary

- Instances: 15
- Customers: between 51 and 1002
- Hotels: 10
- Optimal solutions not known
- New best known: 15

Gap:
- Max: 12.82%
- Min: 1.62%
- Avg: 7.42%
Comparison - Execution time

Figure: Average time
Comparison - Execution time

Figure: Size vs. Time
Contents

1  Problem description
2  Solution technique
3  Computational experiments
4  Conclusions
Conclusions and further research

- Difficult combinatorial optimisation problem
- New number of optimal solutions for Set 2, however, the use of the mathematical formulation is still impractical
- Good results when compared with Gurobi’s optimal solutions and with known-optimal solutions
- Consistently produce better results than the existing heuristics
- Hotel costs, time windows, etc.