A new heuristic algorithm for the Vehicle Routing Problem

Roberto De Franceschi, DEI, University of Padua

Matteo Fischetti, DEI, University of Padua

Paolo Toth, DEIS, University of Bologna

Aussois, January 2004
A method for the TSP (Sarvanov and Doroshko, 1981)

The **ASSIGN** neighborhood

1. consider a **given** tour as a sequence of nodes

2. fix the nodes in **odd** position, and remove the nodes in **even** position

3. Reassign the removed nodes in optimal way—an easy-solvable min-cost assignment problem

Neighborhood of **exponential cardinality** searchable in polynomial time, recently studied by:
- Deineko and Woeginger (2000)
- Firla, Spille and Weismantel (2002)
Capacitated Vehicle Routing Problem

Input

Depot

K vehicles
each with capacity C

N customers
with known demand $d_i$

Goal

K routes
not exceeding the given capacity

with minimum total cost
It seems useful to “move” node v₃ to route Rₐ (assuming this is feasible w.r.t. the capacity constraints)

But … this cannot be done by a simple position-exchange between nodes

Introduce the concepts of restricted solution and insertion point
Basic extensions – Part II

It seems useful to “move” both \( v_3 \) and \( v_4 \) to \( R_A \) (if feasible)

But … this cannot be done in one step by only “moving” single nodes

… solution

go beyond the basic odd/even scheme and introduce the notion of extracted node sequences
Basic extensions – Part III

**Issue ...**

It is not possible to insert both \(v_1\) and \(v_3-v_4\) into the insertion point IP

**... solution**

generate a (possibly large) number of derived sequences through extracted nodes

In the example, it is useful to generate the sequence \(v_1-v_3-v_4\) to be placed in the insertion point IP
# The SERR algorithm

<table>
<thead>
<tr>
<th>Steps</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Initialization</strong></td>
<td>generate, by any heuristic or metaheuristic, an initial solution</td>
</tr>
<tr>
<td><strong>Iteratively:</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Selection</strong></td>
<td>select the nodes to be extracted, according to suitable criteria (schemes)</td>
</tr>
<tr>
<td><strong>Extraction</strong></td>
<td>remove the selected nodes and generate the restricted solution</td>
</tr>
<tr>
<td><strong>Recombination</strong></td>
<td>starting from extracted nodes, generate a (possibly large) number of derived sequences</td>
</tr>
<tr>
<td><strong>Re-insertion</strong></td>
<td>re-insert a subset of the derived sequences into the restricted solution, in such a way that all the extracted nodes are covered again</td>
</tr>
<tr>
<td><strong>Evaluation</strong></td>
<td>verify a stopping condition and return, if it is the case, to the selection step</td>
</tr>
</tbody>
</table>
An example
Node re-insertion is done by solving the following \textit{set-partitioning} model:

\[
\min \sum_{s \in S} \sum_{i \in I} C_{si} x_{si}
\]

\[
\sum_{s \in S} x_{si} = 1 \quad \forall v \text{ extracted}
\]

\[
\sum_{i \in I} x_{si} \leq 1 \quad \forall i \in I
\]

\[
d(r) + \sum_{s \in S} \sum_{i \in r} d(s) x_{si} \leq C \quad \forall r \in R
\]

\[
0 \leq x_{sj} \leq 1 \quad \text{integer} \quad \forall s \in S, \forall i \in I
\]

\(x_{si} = 1\) if and only if sequence \(s\) goes into the insertion point \(i\)

\(C_{si}\) (best) insertion cost of sequence \(s\) into the insertion point \(i\)

\(d(r)\) total demand of the restricted route \(r\)

\(d(s)\) total demand in the node sequence \(s\)
An example (cont.d)
Initial Solution
Interesting solutions

Initial solution: cost 1076
Xu and Kelly, 1996

Final solution: cost 1067
New best known solution
Interesting solutions

Instance M-n151-k12 with rounded costs

Initial solution: cost 1023
Gendreau, Hertz and Laporte, 1996

Final solution: cost 1022
*New best known solution*
### Some Computational Results

<table>
<thead>
<tr>
<th>Instance</th>
<th>Optimal</th>
<th>SERR sol.</th>
<th>Gap</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>P-n50-k8</td>
<td>631</td>
<td>631</td>
<td>0.00%</td>
<td>11:08</td>
</tr>
<tr>
<td>P-n55-k10</td>
<td>694</td>
<td>700</td>
<td>0.86%</td>
<td>16:50</td>
</tr>
<tr>
<td>P-n60-k10</td>
<td>744</td>
<td>744</td>
<td>0.00%</td>
<td>25:01</td>
</tr>
<tr>
<td>P-n60-k15</td>
<td>968</td>
<td>975</td>
<td>0.72%</td>
<td>12:27</td>
</tr>
<tr>
<td>P-n65-k10</td>
<td>792</td>
<td>796</td>
<td>0.51%</td>
<td>12:26</td>
</tr>
<tr>
<td>P-n70-k10</td>
<td>827</td>
<td>834</td>
<td>0.48%</td>
<td>50:08</td>
</tr>
<tr>
<td>B-n68-k9</td>
<td>1272</td>
<td>1275</td>
<td>0.24%</td>
<td>3:02:01</td>
</tr>
<tr>
<td>E-n51-k5</td>
<td>521</td>
<td>521</td>
<td>0.00%</td>
<td>4:30</td>
</tr>
<tr>
<td>E-n76-k7</td>
<td>682</td>
<td>682</td>
<td>0.00%</td>
<td>27:35</td>
</tr>
<tr>
<td>E-n76-k8</td>
<td>735</td>
<td>742</td>
<td>0.95%</td>
<td>30:39</td>
</tr>
<tr>
<td>E-n76-k10</td>
<td>830</td>
<td>835</td>
<td>0.60%</td>
<td>1:19:30</td>
</tr>
<tr>
<td>E-n76-k14</td>
<td>1021</td>
<td>1032</td>
<td>1.08%</td>
<td>2:45:20</td>
</tr>
<tr>
<td>E-n101-k8</td>
<td>815</td>
<td>820</td>
<td>0.61%</td>
<td>2:54:04</td>
</tr>
<tr>
<td>E051-05e</td>
<td>524.61</td>
<td>524.61</td>
<td>0.00%</td>
<td>4:51</td>
</tr>
<tr>
<td>E076-10e</td>
<td>835.26</td>
<td>835.32</td>
<td>&lt; 0.01%</td>
<td>1:12:05</td>
</tr>
<tr>
<td>E101-08e</td>
<td>826.14</td>
<td>831.91</td>
<td>0.70%</td>
<td>2:30:55</td>
</tr>
<tr>
<td>E101-10c</td>
<td>819.56</td>
<td>819.56</td>
<td>0.00%</td>
<td>2:35:36</td>
</tr>
<tr>
<td>E-n101-k14</td>
<td>-</td>
<td>1076 -&gt; 1067</td>
<td>-</td>
<td>1:36:05</td>
</tr>
<tr>
<td>M-n151-k12-a</td>
<td>-</td>
<td>1023 -&gt; 1022</td>
<td>-</td>
<td>7:46:33</td>
</tr>
</tbody>
</table>

CPU times in the format [hh:]mm:ss  
PC: Pentium M 1.6GHz

(*) Most optimal solutions have been found very recently by Fukasawa, Poggi de Aragao, Reis, and Uchoa (September 2003)
Results

Convergence properties of the SERR method

Low-cost solutions available in the first iterations

The best heuristics from the literature are credited for errors of about 2%
Conclusions

Achieved goals

1. **Definition** of a new *neighborhood* with exponential cardinality and of an effective (non-polynomial) *search algorithm*
2. **Simple implementation** based on a general ILP solver
3. **Evaluation** of the algorithm on a widely-used set of instances
4. Determination of the **new best solution** for two of the few instances not yet solved to optimality

Future directions of work

1. **Adaptation** of the method to more constrained versions of VRP, including VRP with *precedence constraints*
2. Use of an external *metaheuristic scheme*
Special contents…
Selected literature on VRP heuristics

1959  Dantzig and Ramser: problem formulation
1964  Clarke and Wright: heuristic algorithm
       Balinski and Quandt: set-partitioning model
1976  Foster and Ryan: Petal heuristic
1981  Fisher and Jaikumar: Generalized Assignment heuristic
1993  Taillard: Tabu Search metaheuristic
1998  Toth and Vigo: Granular Tabu Search metaheuristic

Properties

- Important practical applications
- NP-hard
- Generalizes the Traveling Salesman Problem (TSP)