Recent advances in Variable neighborhood search

Nenad Mladenović,

Department of Mathematics, Brunel University - London UK

- Introduction
- Variable neighborhood search algorithms
- Recent advances in VNS
  - Global Continuous VNS
  - Polynomial neighborhoods for discrete problems
  - 2-phase VNS
  - 3-level VNS
  - Formulation space VNS
  - Matheuristics and VNS
  - VNS Hybrids

Introduction

• Mladenovic (1995) - Variable neighborhood algorithm -
a new metaheuristic for combinatorial optimization.
• This paper cited around 1,300 times (Google scholar).
• This paper cited > 500 times (Web of Knowledge)
• Euro Mini Conference devoted to VNS (2005);
• 3 VNS special issues (JOH, EJOR and IMA-MAN).
• Several book Chapters.

• GOW-2012, Natal, Brazil, Jun 26 - 29, 2012.
Variable metric algorithm

Assume that the function $f(x)$ is approximated by its Taylor series

$$f(x) = \frac{1}{2}x^TAx - b^Tx$$

$$x_{i+1} - x_i = -H_{i+1}(\nabla f(x_{i+1}) - \nabla f(x_i)).$$

Function $\text{VarMetric}(x)$

let $x \in \mathbb{R}^n$ be an initial solution

$H \leftarrow I; \ g \leftarrow -\nabla f(x)$

for $i = 1$ to $n$ do

$$\alpha^* \leftarrow \text{arg min}_\alpha f(x + \alpha \cdot Hg)$$

$$x \leftarrow x + \alpha^* \cdot Hg$$

$$g \leftarrow -\nabla f(x)$$

$H \leftarrow H + U$

end
Variable neighborhood search

- Let $\mathcal{N}_k$, $(k = 1, \ldots, k_{max})$, a finite set of pre-selected neighborhood structures,
- $\mathcal{N}_k(x)$ the set of solutions in the $k^{th}$ neighborhood of $x$.
- Most local search heuristics use only one neighborhood structure, i.e., $k_{max} = 1$.
- An optimal solution $x_{opt}$ (or global minimum) is a feasible solution where a minimum is reached.
- We call $x' \in X$ a local minimum with respect to $\mathcal{N}_k$ (w.r.t. $\mathcal{N}_k$ for short), if there is no solution $x \in \mathcal{N}_k(x') \subseteq X$ such that $f(x) < f(x')$.
- Metaheuristics (based on local search procedures) try to continue the search by other means after finding the first local minimum. VNS is based on three simple facts:
  - A local minimum w.r.t. one neighborhood structure is not necessarily so for another;
  - A global minimum is a local minimum w.r.t. all possible neighborhood structures;
  - For many problems, local minima w.r.t. one or several $\mathcal{N}_k$ are relatively close to each other.
Variable neighborhood search

• In order to solve optimization problem by using several neighborhoods, facts 1 to 3 can be used in three different ways:
  ▶ (i) deterministic;
  ▶ (ii) stochastic;
  ▶ (iii) both deterministic and stochastic.

• Some VNS variants
  ▶ Variable neighborhood descent (VND) (sequential, nested)
  ▶ Reduced VNS (RVNS)
  ▶ Basic VNS (BVNS)
  ▶ Skewed VNS (SVNS)
  ▶ General VNS (GVNS)
  ▶ VN Decomposition Search (VNDS)
  ▶ Parallel VNS (PVNS)
  ▶ Primal Dual VNS (P-D VNS)
  ▶ Reactive VNS
  ▶ Backward-Forward VNS;
  ▶ Exterior point VNS;
  ▶ VN Branching
  ▶ VN Pump and VN Diving;
  ▶ Continuous VNS
  ▶ Mixed Nonlinear VNS (RECIPE), etc.

• GOW-2012, Natal, Brazil, Jun 26 - 29, 2012.
Continuous global optimization problem
\[ \min_{x \in X \subseteq \mathbb{R}^n} f(x) \]
where \( f : \mathbb{R}^n \rightarrow \mathbb{R} \) is a continuous function.

- No further assumptions are made on \( f \). In particular \( f \) does not need to be convex or smooth.
- Existing VNS methods:
  - Hansen, Ml. 2001, Bilinear (Pooling) problem
  - Mladenovic et al 2003, Radar design
  - Dražić, Ml et al 2005, Unconstrained problems.
  - Drazic, Liberti 2005, SNOPT used as a local search
  - Toskari, Guner 2007, No metric distances used
  - Mladenovic et al 2008, Unconstrained and Constrained
  - Audet et al 2008, Hybrid with Trust region
  - Bierlaire et al 2010, VNS with trust region method as a local search
  - Carrizosa et al 2012, Gaussian VNS.

Algorithm Glob-VNS

/* Initialization */
01 Select the pairs \((G_l, P_l)\), \(l = 1, \ldots, m\) of geometry structures and
distribution types and a set of radii \(\rho_i, i = 1, \ldots, k_{\text{max}}\)
02 Choose an arbitrary initial point \(x \in S\)
03 Set \(x^* \leftarrow x, f^* \leftarrow f(x)\)

/* Main loop */
04 repeat the following steps until the stopping condition is met
05 Set \(l \leftarrow 1\)
06 repeat the following steps until \(l > m\)
07 Form the neighborhoods \(N_k, k = 1, \ldots, k_{\text{max}}\) using
geometry structure \(G_l\) and radii \(\rho_k\)
08 Set \(k \leftarrow 1\)
09 repeat the following steps until \(k > k_{\text{max}}\)
10 Shake: Generate at random a point \(y \in N_k(x^*)\) using
random distribution \(P_l\)
11 Apply some local search method from \(y\) to obtain a local
minimum \(y'\)
12 if \(f(y') < f^*\) then
13 Set \(x^* \leftarrow y', f^* \leftarrow f(y')\) and goto line 05
14 endif
15 Set \(k \leftarrow k + 1\)
16 end
17 Set \(l \leftarrow l + 1\)
18 end
19 Stop. Point \(x^*\) is an approximate solution of the problem.
Algorithm Gauss-VNS

/* Initialization */
01 Select the set of covariance matrices $\Sigma_k$, $k = 1, \ldots, k_{\text{max}}$
02 Choose an arbitrary initial point $x \in S$
03 Set $x^* \leftarrow x$, $f^* \leftarrow f(x)$

/* Main loop */
04 repeat the following steps until the stopping condition is met
   Set $k \leftarrow 1$
   repeat the following steps until $k > k_{\text{max}}$
      Shake: Generate $y$ from a Gaussian distribution with
              mean $x^*$ and covariance matrix $\Sigma_k$
      Apply some local search method from $y$ to obtain a local
              minimum $y'$
      if $f(y') < f^*$ then
         Set $x^* \leftarrow y'$, $f^* \leftarrow f(y')$ and goto line 05
      endif
      Set $k \leftarrow k + 1$
   end
05 Stop. Point $x^*$ is an approximate solution of the problem.
Continous VNS

<table>
<thead>
<tr>
<th>function</th>
<th>$n$</th>
<th>local minim.</th>
<th>$k_{\text{max}}$</th>
<th>Computer effort</th>
<th>% deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Glob-VNS</td>
<td>Gauss-VNS</td>
</tr>
<tr>
<td>AC10</td>
<td>10</td>
<td>SD</td>
<td>10</td>
<td>188670</td>
<td>50149</td>
</tr>
<tr>
<td>AC20</td>
<td>20</td>
<td>SD</td>
<td>10</td>
<td>433194</td>
<td>158412</td>
</tr>
<tr>
<td>AC30</td>
<td>30</td>
<td>SD</td>
<td>10</td>
<td>909918</td>
<td>304825</td>
</tr>
<tr>
<td>AC40</td>
<td>40</td>
<td>SD</td>
<td>10</td>
<td>1577138</td>
<td>528718</td>
</tr>
<tr>
<td>AC50</td>
<td>50</td>
<td>SD</td>
<td>10</td>
<td>4791075</td>
<td>1143721</td>
</tr>
<tr>
<td>AC60</td>
<td>60</td>
<td>SD</td>
<td>10</td>
<td>7820247</td>
<td>2315178</td>
</tr>
<tr>
<td>AC70</td>
<td>70</td>
<td>SD</td>
<td>10</td>
<td>36641634</td>
<td>4255533</td>
</tr>
<tr>
<td>AC80</td>
<td>80</td>
<td>SD</td>
<td>10</td>
<td>212944367</td>
<td>17180658</td>
</tr>
</tbody>
</table>

Average: 412.96%

Table 1: Ackley function
<table>
<thead>
<tr>
<th>function</th>
<th>$n$</th>
<th>local minim.</th>
<th>$k_{\text{max}}$</th>
<th>Computer effort</th>
<th>% deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Glob-VNS</td>
<td>Gauss-VNS</td>
</tr>
<tr>
<td>MPE10</td>
<td>10</td>
<td>SD</td>
<td>10</td>
<td>8102</td>
<td>5015</td>
</tr>
<tr>
<td>MPE20</td>
<td>20</td>
<td>SD</td>
<td>10</td>
<td>26647</td>
<td>21172</td>
</tr>
<tr>
<td>MPE30</td>
<td>30</td>
<td>SD</td>
<td>10</td>
<td>66441</td>
<td>49162</td>
</tr>
<tr>
<td>MPE40</td>
<td>40</td>
<td>SD</td>
<td>10</td>
<td>118006</td>
<td>109468</td>
</tr>
<tr>
<td>MPE50</td>
<td>50</td>
<td>SD</td>
<td>10</td>
<td>202280</td>
<td>143309</td>
</tr>
<tr>
<td>MPE100</td>
<td>100</td>
<td>SD</td>
<td>10</td>
<td>830343</td>
<td>1183873</td>
</tr>
<tr>
<td>MPE150</td>
<td>150</td>
<td>SD</td>
<td>10</td>
<td>2353315</td>
<td>2802372</td>
</tr>
<tr>
<td>MPE200</td>
<td>200</td>
<td>SD</td>
<td>10</td>
<td>7683209</td>
<td>5859705</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 2: Molecular potential energy function
<table>
<thead>
<tr>
<th>function</th>
<th>$n$</th>
<th>local minim.</th>
<th>$k_{\text{max}}$</th>
<th>Computer effort</th>
<th>% deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Glob-VNS</td>
<td>Gauss-VNS</td>
</tr>
<tr>
<td>RA10</td>
<td>10</td>
<td>SD</td>
<td>15</td>
<td>52471</td>
<td>85589</td>
</tr>
<tr>
<td>RA20</td>
<td>20</td>
<td>SD</td>
<td>15</td>
<td>213597</td>
<td>287075</td>
</tr>
<tr>
<td>RA30</td>
<td>30</td>
<td>SD</td>
<td>15</td>
<td>366950</td>
<td>599635</td>
</tr>
<tr>
<td>RA40</td>
<td>40</td>
<td>SD</td>
<td>15</td>
<td>697160</td>
<td>1115923</td>
</tr>
<tr>
<td>RA50</td>
<td>50</td>
<td>SD</td>
<td>15</td>
<td>1334842</td>
<td>1504701</td>
</tr>
<tr>
<td>RA100</td>
<td>100</td>
<td>SD</td>
<td>15</td>
<td>5388075</td>
<td>6248753</td>
</tr>
<tr>
<td>RA150</td>
<td>150</td>
<td>SD</td>
<td>15</td>
<td>11007093</td>
<td>13678014</td>
</tr>
<tr>
<td>RA200</td>
<td>200</td>
<td>SD</td>
<td>15</td>
<td>24026456</td>
<td>31639001</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 3: Rastrigin function
New deterministic neighborhoods within VND

Function Seq-VND\((x, \ell_{max})\)
\[
\ell \leftarrow 1 \quad \text{// Neighborhood counter}
\]
\[
\text{repeat}
\]
\[
\quad i \leftarrow 0 \quad \text{// Neighbor counter}
\]
\[
\quad \text{repeat}
\]
\[
\quad \quad i \leftarrow i + 1
\]
\[
\quad \quad x' \leftarrow \text{arg min}\{f(x), f(x_i)\}, x_i \in N_\ell(x) \quad \text{// Compare}
\]
\[
\quad \text{until } (f(x') < f(x) \text{ or } i = |N_\ell(x)|)
\]
\[
\ell, x \leftarrow \text{NeighborhoodChange} (x, x', \ell); \quad \text{// Neighborhood change}
\]
\[
\text{until } \ell = \ell_{max}
\]

- The final solution of Seq-VND should be a local minimum with respect to all \(\ell_{max}\) neighborhoods.
- The chances to reach a global minimum are larger than with a single neighborhood structure.
- The total size of Seq-VND is equal to the union of all neighborhoods used.
• If neighborhoods are disjoint (no common element in any two) then the following holds

\[ |\mathcal{N}_{\text{Seq-VND}}(x)| \leq \sum_{\ell=1}^{\ell_{\text{max}}} |\mathcal{N}_\ell(x)|, \ x \in X. \]
Nested VND

• Assume that we define two neighborhood structures \((\ell_{\text{max}} = 2)\). In the nested VND we in fact perform local search with respect to the first neighborhood in any point of the second.

• The cardinality of neighborhood obtained with the nested VND is product of cardinalities of neighborhoods included, i.e.,

\[
|\mathcal{N}_{\text{Nest-VND}}(x)| \leq \prod_{\ell=1}^{\ell_{\text{max}}} |\mathcal{N}_\ell(x)|, \; x \in X.
\]

• The pure Nest-VND neighborhood is much larger than the sequential.

• The number of local minima w.r.t. Nest-VND will be much smaller than the number of local minima w.r.t. Seq-VND.
Nested VND

Function Nest-VND \((x, x', k)\)

Make an order of all \(\ell_{max} \geq 2\) neighborhoods that will be used in the search.

Find an initial solution \(x\); let \(x_{opt} = x\), \(f_{opt} = f(x)\)

Set \(\ell = \ell_{max}\)

repeat

\[
\begin{align*}
\text{if} & \text{ all solutions from } \ell \text{ neighborhood are visited then } \ell = \ell + 1 \\
\text{if} & \text{ there is any non visited solution } x_{\ell} \in N_{\ell}(x) \text{ and } \ell \geq 2 \text{ then } x_{cur} = x_{\ell}, \ell = \ell - 1 \\
\text{if} & \ell = 1 \text{ then} \\
& \text{Find objective function value } f = f(x_{cur}) \\
& \quad \text{if } f < f_{opt} \text{ then } x_{opt} = x_{cur}, f_{opt} = f_{cur}
\end{align*}
\]

until \(\ell = \ell_{max} + 1\) (i.e., until there is no more points in the last neighborhood)
After exploring $b$ (a parameter) neighborhoods, we switch from a nested to a sequential strategy. We can interrupt nesting at some level $b$ ($1 \leq b \leq \ell_{max}$) and continue with the list of the remaining neighborhoods in sequential manner.

- If $b = 1$, we get Seq-VND. If $b = \ell_{max}$ we get Nest-VND.
- Since nested VND intensifies the search in a deterministic way, boost parameter $b$ may be seen as a balance between intensification and diversification in deterministic local search with several neighborhoods.
- Its cardinality is clearly

$$|\mathcal{N}_{\text{Mix-VND}}(x)| \leq \sum_{\ell=b}^{\ell_{max}} |\mathcal{N}_\ell(x)| \times \prod_{\ell=1}^{b-1} |\mathcal{N}_\ell(x)|, \ x \in X.$$
<table>
<thead>
<tr>
<th>$n$</th>
<th>$Q$</th>
<th>Local Search</th>
<th>Min. % dev</th>
<th>Max. % dev</th>
<th>Avg. % dev</th>
<th>Avg. time</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>10</td>
<td>forward-insertion</td>
<td>95.650</td>
<td>255.517</td>
<td>181.195</td>
<td>0.088</td>
</tr>
<tr>
<td></td>
<td></td>
<td>backward-insertion</td>
<td>94.753</td>
<td>252.827</td>
<td>186.318</td>
<td>0.081</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2–opt</td>
<td>13.882</td>
<td>242.910</td>
<td>32.433</td>
<td>0.138</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Seq–VND</td>
<td>12.275</td>
<td>242.910</td>
<td>27.808</td>
<td>0.163</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Seq–VND–3</td>
<td>8.991</td>
<td>242.910</td>
<td>24.309</td>
<td>0.478</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Mix–VND</td>
<td>1.269</td>
<td>242.910</td>
<td>12.958</td>
<td>2.989</td>
</tr>
<tr>
<td>400</td>
<td>10</td>
<td>forward-insertion</td>
<td>78.320</td>
<td>218.770</td>
<td>165.603</td>
<td>0.385</td>
</tr>
<tr>
<td></td>
<td></td>
<td>backward-insertion</td>
<td>81.881</td>
<td>218.770</td>
<td>169.416</td>
<td>0.317</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2–opt</td>
<td>12.078</td>
<td>217.104</td>
<td>21.852</td>
<td>0.831</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Seq–VND</td>
<td>10.684</td>
<td>217.104</td>
<td>18.954</td>
<td>0.769</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Mix–VND</td>
<td>1.492</td>
<td>217.104</td>
<td>6.573</td>
<td>26.569</td>
</tr>
</tbody>
</table>

Table 4: Comparison of different local search algorithms on two instances

3-level VNS

- Our basic method is Skewed Variable neighborhood search (SVNS)
- SVNS is the VNS variant that allows moves to slightly worse solutions, but only if they are far from the incumbent.

Function \text{3L-VNS} (\( x, k_{max}, t_{max}, \alpha \))

\[\begin{align*}
\text{repeat} \\
\ k &\leftarrow 1; x_{\text{best}} \leftarrow x \\
\text{repeat} \\
\ x' &\leftarrow \text{Shake}(x, k) \quad /* \text{Shaking} */ \\
\ x'' &\leftarrow \text{VNDS}(x') \quad /* \text{VNDS Local search} */ \\
\text{if} \ f(x'') < f(x_{\text{best}}) \text{ then} \ x_{\text{best}} \leftarrow x'' \\
\text{NeighborhoodChangeS}(x, x'', x_{\text{best}}, k, \alpha) \quad /* \text{Skewed move} */ \\
\text{until} \ k = k_{max}; \\
\text{t} &\leftarrow \text{CpuTime()} \\
\text{until} \ t > t_{max};
\end{align*}\]

- 3L-VNS is a Skewed VNS, with VNDS used instead of a local search
- The distance function \( \rho(x, y) \) measures the number of different assignment in \( x \) and \( y \).

\[\begin{align*}
\text{GOW-2012, Natal, Brazil, Jun 26 - 29, 2012.}
\end{align*}\]
Formulation space search (FSS)

- Reformulation descent for Circle packing (MI et al 2005)
- FSS for circle packing (MI et al 2005, 2007);
- Kochetov for time tabling (2006);
- Variable space search (Hertz, Zuferley 2007)
- FSS for circle packing (Beasley 2011)
- Variable Objective Search for maximum Independent set (Butenko 2012).
- Discrete - Continuous reformulation (Brimberg et al 2012)
- Variable formulation search (Prado et al 2012)
Discrete - Continuous reformulation (Brimberg et al 2012)

Location-Allocation problem

- The continuous location-allocation problem, also referred to as the multi-source Weber problem, is one of the basic models in location theory.
- The objective is to generate optimal sites in continuous space, notably $\mathbb{R}^2$, for $m$ new facilities in order to minimize a sum of transportation (or service) costs to a set of $n$ fixed points or customers with known demands.
- The problem in its most basic form, which will be considered herein, makes the following assumptions:
  - there are no interactions between the new facilities;
  - the number of new facilities ($m$) is given;
  - the cost function is a weighted sum of the Euclidean distances between new facilities and fixed points, where the weights are proportional to the flow or interaction between the corresponding pairs of locations;
  - the new facilities have infinite capacities.
LA problem formulation

\[
\min_{W,X} \sum_{i=1}^{n} \sum_{j=1}^{m} w_{ij} \|x_j - a_i\| \\
\text{s.t.} \quad \sum_{j=1}^{m} w_{ij} = w_i, \quad i = 1, \ldots, n, \\
\quad w_{ij} \geq 0, \quad \forall i, j,
\]

- \(a_i = (a_{i1}, a_{i2})\) is the known location of customer \(i, i = 1, \ldots, n;\)
- \(X = (x_1, \ldots, x_m)\) denotes the matrix of location decision variables, with \(x_j = (x_{j1}, x_{j2})\) being the unknown location of facility \(j, j = 1, \ldots, m;\)
- \(w_i\) is the given total demand or flow required by customer \(i, i = 1, \ldots, n;\)
- \(W = (w_{ij})\) denotes the vector of allocation decision variables, where \(w_{ij}\) gives the flow to customer \(i\) from facility \(j, i = 1, \ldots, n, \quad j = 1, \ldots, m;\)
- \(\|x_j - a_i\| = [(x_{j1} - a_{i1})^2 + (x_{j2} - a_{i2})^2]^{1/2}\) is the Euclidean norm.
Heuristics for solving LA problem

- Cooper (1964) suggested several heuristics: ALT, p-median, ...
- Love and Juel (1983) proposed 4 heuristics (H1-H4)
- Brimberg and Mladenovic (1995) Tabu search
- Brimberg and Mladenovic (1996) VNA
- Brimberg et al. (2000) compared almost 20 different heuristics, including several new (GA, VNS, etc.)
- Salhi, Gamal (2003) - GA
- Jabalameli, Ghaderi (2008) Hybrid Memetic and VNS.
FSS local search for LA problem

- **Step 1.** Using random initial solution and Cooper’s alternate heuristic, find local minimum $x_{opt}$.

- **Step 2.** Find a set of unoccupied points $U$, i.e., new facilities from $x_{opt}$ that do not coincide with the current set of fixed points. 

- **Step 3.** Add unoccupied facilities obtained in Step 2 to the set of fixed points ($n := n + \text{card}(U)$) and solve the related $m$-Median problem. Denote $m$-median solution with $x_{med}$. 

- **Step 4.** If $f(x_{med}) = f(x_{opt})$, stop.
  Otherwise, return to Step 1 with random initial solution $= x_{med}$, if $n < 1.4|V|$, or with new random initial solution and $n = |V|$. 

---

- **GOW-2012, Natal, Brazil, Jun 26 - 29, 2012.**
<table>
<thead>
<tr>
<th>$m$</th>
<th>Objective function values</th>
<th>% deviation</th>
<th>CPU Times (sec.)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Old best</td>
<td>VNS</td>
<td>FSS</td>
</tr>
<tr>
<td>5</td>
<td>1851879.88</td>
<td>1851879.88</td>
<td>1851877.25</td>
</tr>
<tr>
<td>10</td>
<td>1249564.75</td>
<td>1249564.75</td>
<td>1249564.75</td>
</tr>
<tr>
<td>15</td>
<td>980132.13</td>
<td>980146.25</td>
<td>980131.69</td>
</tr>
<tr>
<td>20</td>
<td>828802.00</td>
<td>828696.00</td>
<td>828685.69</td>
</tr>
<tr>
<td>25</td>
<td>722061.19</td>
<td>722053.56</td>
<td>722000.69</td>
</tr>
<tr>
<td>30</td>
<td>638263.00</td>
<td>638236.00</td>
<td>638212.50</td>
</tr>
<tr>
<td>35</td>
<td>577526.63</td>
<td>577523.94</td>
<td>577501.06</td>
</tr>
<tr>
<td>40</td>
<td>529866.19</td>
<td>529755.75</td>
<td>529680.81</td>
</tr>
<tr>
<td>45</td>
<td>489650.00</td>
<td>489551.00</td>
<td>489596.78</td>
</tr>
<tr>
<td>50</td>
<td>453164.00</td>
<td>453175.88</td>
<td>453149.09</td>
</tr>
<tr>
<td>55</td>
<td>422770.00</td>
<td>422903.06</td>
<td>422647.75</td>
</tr>
<tr>
<td>60</td>
<td>397784.41</td>
<td>397759.50</td>
<td>397745.47</td>
</tr>
<tr>
<td>65</td>
<td>376759.50</td>
<td>376740.81</td>
<td>376630.47</td>
</tr>
<tr>
<td>70</td>
<td>357385.00</td>
<td>357363.25</td>
<td>357335.34</td>
</tr>
<tr>
<td>75</td>
<td>340242.00</td>
<td>340208.19</td>
<td>340168.03</td>
</tr>
<tr>
<td>80</td>
<td>326053.19</td>
<td>326107.28</td>
<td>326035.28</td>
</tr>
</tbody>
</table>

Table 5: Comparison of VNS and FSS with $t_{max} = 300$ seconds, $k_{max} = p$ for both.

<table>
<thead>
<tr>
<th>m</th>
<th>Objective function values</th>
<th>% deviation</th>
<th>CPU Times (sec.)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Old best</td>
<td>VNS</td>
<td>FSS</td>
</tr>
<tr>
<td>85</td>
<td>313738.19</td>
<td>313769.16</td>
<td>313670.47</td>
</tr>
<tr>
<td>90</td>
<td>302837.00</td>
<td>302816.50</td>
<td>302565.94</td>
</tr>
<tr>
<td>95</td>
<td>292875.09</td>
<td>292782.72</td>
<td>292730.59</td>
</tr>
<tr>
<td>100</td>
<td>283113.00</td>
<td>282964.00</td>
<td>282890.41</td>
</tr>
<tr>
<td>105</td>
<td>274576.00</td>
<td>274026.78</td>
<td>273705.91</td>
</tr>
<tr>
<td>110</td>
<td>265801.00</td>
<td>265329.50</td>
<td>265292.34</td>
</tr>
<tr>
<td>115</td>
<td>257605.00</td>
<td>257397.05</td>
<td>257243.98</td>
</tr>
<tr>
<td>120</td>
<td>249584.00</td>
<td>249532.33</td>
<td>249409.92</td>
</tr>
<tr>
<td>125</td>
<td>242930.00</td>
<td>242599.98</td>
<td>242111.36</td>
</tr>
<tr>
<td>130</td>
<td>236154.00</td>
<td>235945.00</td>
<td>235573.94</td>
</tr>
<tr>
<td>135</td>
<td>230431.00</td>
<td>229400.88</td>
<td>229169.05</td>
</tr>
<tr>
<td>140</td>
<td>224504.00</td>
<td>223530.78</td>
<td>223423.50</td>
</tr>
<tr>
<td>145</td>
<td>218279.00</td>
<td>218441.47</td>
<td>217791.58</td>
</tr>
<tr>
<td>150</td>
<td>212926.00</td>
<td>212404.06</td>
<td>212344.36</td>
</tr>
<tr>
<td>Aver.</td>
<td>471575.19</td>
<td>471420.19</td>
<td>471315.09</td>
</tr>
</tbody>
</table>

Table 6: Comparison of VNS and FSS with $t_{max} = 300$ seconds, $k_{max} = p$ for both.
The aim of VFS is to determine whether a given solution is more promising than other to continue the search, beyond the value of the objective function.

This fact is specially helpful for many min-max problems,

In this case, when two solutions have same value of the objective function, VFS performs a new comparison based on the use of alternative formulations of the problem.

Function \texttt{Accept} \((x, x', p)\)

\begin{verbatim}
for i = 0, p do

    condition1 = \(f_i(x') < f_i(x)\)
    condition2 = \(f_i(x') > f_i(x)\)

    if condition1 then
        return True
    else
        if condition2 then
            return False
        end
    end
end
\end{verbatim}
Variable Formulation Search - Min Cutwidth problem

Figure 1: (a) Graph $G$ with six vertices and nine edges. (b) Ordering $f$ of the vertices of the graph in (a) with the corresponding cutwidth of each vertex.
### VFS - Min Cutwidth problem

<table>
<thead>
<tr>
<th></th>
<th>BVNS</th>
<th>VFS(_1)</th>
<th>VFS(_2)</th>
<th>VFS(_3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Avg.</td>
<td>137.31</td>
<td>93.56</td>
<td>91.56</td>
<td>90.75</td>
</tr>
<tr>
<td>Dev. (%)</td>
<td>192.44</td>
<td>60.40</td>
<td>49.23</td>
<td>48.22</td>
</tr>
<tr>
<td>CPUt (s)</td>
<td>30.17</td>
<td>30.47</td>
<td>30.50</td>
<td>30.96</td>
</tr>
</tbody>
</table>

Table 7: Impact of the use of alternative formulations in the search process.

- 30 seconds for each instance of the *Test* data set.
- \(K_{max} = 0.1n\) and they start from the same random solution.

<table>
<thead>
<tr>
<th></th>
<th>HB (87)</th>
<th>GPR</th>
<th>SA</th>
<th>SS</th>
<th>VFS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Avg.</td>
<td>364.83</td>
<td>346.21</td>
<td>315.22</td>
<td>314.39</td>
<td></td>
</tr>
<tr>
<td>Dev. (%)</td>
<td>95.13</td>
<td>53.30</td>
<td>3.40</td>
<td>1.77</td>
<td></td>
</tr>
<tr>
<td>#Best</td>
<td>2</td>
<td>8</td>
<td>47</td>
<td>61</td>
<td></td>
</tr>
<tr>
<td>CPUt (s)</td>
<td>557.49</td>
<td>435.40</td>
<td>430.57</td>
<td>128.12</td>
<td></td>
</tr>
</tbody>
</table>

Table 8: Comparison with the state-of-the-art algorithms over the Harwell-Boeing data set.
Thank you for your attention!

nenad.mladenovic@brunel.ac.uk