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.
- Mladenovic, Hansen (1997) Variable neighborhood search.
- 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.

Variable metric algorithm

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

$$f(x) = \frac{1}{2}x^T Ax - b^T x$$

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

Function `VarMetric`(x)
let $x \in R^n$ be an initial solution
$H \leftarrow I$; $g \leftarrow -\nabla f(x)$
for $i = 1$ `to` n **do**

  $\alpha^* \leftarrow arg\min_\alpha f(x + \alpha \cdot Hg)$
  $x \leftarrow x + \alpha^* \cdot Hg$
  $g \leftarrow -\nabla f(x)$
  $H \leftarrow H + U$
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.

Continuous VNS

- Continuous global optimization problem
  $\min_{x \in X \subseteq R^n} f(x)$
  where $f : R^n \longrightarrow R$ 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 $(\mathcal{G}_l, \mathcal{P}_l),\ l = 1, \ldots, m$ of geometry structures and distribution types and a set of radii $\rho_i,\ i = 1, \ldots, k_{\max}$

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 $\mathcal{N}_k,\ k = 1, \ldots, k_{\max}$ using geometry structure $\mathcal{G}_l$ and radii ρ_k

08   Set $k \leftarrow 1$

09   repeat the following steps until $k > k_{\max}$

10   Shake: Generate at random a point $y \in \mathcal{N}_k(x^*)$ using random distribution $\mathcal{P}_l$

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 end

20 Stop. Point x^* is an approximate solution of the problem

Gauss VNS

Algorithm Gauss-VNS

/* Initialization */

01 Select the set of covariance matrices Σ_k, $k = 1, \ldots, k_{\max}$

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 $k \leftarrow 1$

06   repeat the following steps until $k > k_{\max}$

07   Shake: Generate y from a Gaussian distribution with mean x^* and covariance matrix Σ_k

08   Apply some local search method from y to obtain a local minimum y'

09   if $f(y') < f^*$ then

10   Set $x^* \leftarrow y'$, $f^* \leftarrow f(y')$ and goto line 05

11   endif

12   Set $k \leftarrow k + 1$

13   end

14 end

15 Stop. Point x^* is an approximate solution of the problem.

function	n	local minim.	$k_{\max}$	Computer effort Glob-VNS	Computer effort Gauss-VNS	% deviation
AC10	10	SD	10	188670	50149	276.22%
AC20	20	SD	10	433194	158412	173.46%
AC30	30	SD	10	909918	304825	198.51%
AC40	40	SD	10	1577138	528718	198.30%
AC50	50	SD	10	4791075	1143721	318.90%
AC60	60	SD	10	7820247	2315178	237.78%
AC70	70	SD	10	36641634	4255533	761.04%
AC80	80	SD	10	212944367	17180658	1139.44%
					Average	412.96%

Table 1: Ackley function

function	n	local minim.	$k_{\max}$	Computer effort `Glob-VNS`	Computer effort `Gauss-VNS`	% deviation
MPE10	10	SD	10	8102	5015	61.56%
MPE20	20	SD	10	26647	21172	25.86%
MPE30	30	SD	10	66441	49162	35.15%
MPE40	40	SD	10	118006	109468	7.80%
MPE50	50	SD	10	202280	143309	41.15%
MPE100	100	SD	10	830343	1183873	-29.86%
MPE150	150	SD	10	2353315	2802372	-16.02%
MPE200	200	SD	10	7683209	5859705	31.12%
					Average	19.59%

Table 2: Molecular potential energy function

function	n	local minim.	$k_{\max}$	Computer effort Glob-VNS	Gauss-VNS	% deviation
RA10	10	SD	15	52471	85589	-38.69%
RA20	20	SD	15	213597	287075	-25.60%
RA30	30	SD	15	366950	599635	-38.80%
RA40	40	SD	15	697160	1115923	-37.53%
RA50	50	SD	15	1334842	1504701	-11.29%
RA100	100	SD	15	5388075	6248753	-13.77%
RA150	150	SD	15	11007093	13678014	-19.53%
RA200	200	SD	15	24026456	31639001	-24.06%
					Average	-26.16%

Table 3: Rastrigin function

New deterministic neighborhoods within VND

```
Function Seq-VND(x, ℓ_max)
ℓ ← 1                                // Neighborhood counter
repeat
    i ← 0                            // Neighbor counter
    repeat
        i ← i + 1
        x' ← arg min{f(x), f(x_i)}, x_i ∈ N_ℓ(x) // Compare
    until (f(x') < f(x) or i = |N_ℓ(x)|)

    ℓ, x ← NeighborhoodChange (x, x', ℓ);   // Neighborhood change
until ℓ = ℓ_max
```

- The final solution of Seq-VND should be a local minimum with respect to all ℓ_{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

$$|\mathcal{N}_{\texttt{Seq-VND}}(x)| \leq \sum_{\ell=1}^{\ell_{max}} |\mathcal{N}_\ell(x)|, \; x \in X.$$

Nested VND

- Assume that we define two neighborhood structures ($\ell_{max} = 2$). 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.,

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

- 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

  if all solutions from ℓ neighborhood are visited **then** $\ell = \ell + 1$
  if there is any non visited solution $x_\ell \in N_\ell(x)$ and $\ell \geq 2$ **then** $x_{cur} = x_\ell$, $\ell = \ell - 1$
  if $\ell = 1$ **then**

    Find objective function value $f = f(x_{cur})$
    if $f < f_{opt}$ **then** $x_{opt} = x_{cur}$, $f_{opt} = f_{cur}$

until $\ell = \ell_{max} + 1$ (i.e., until there is no more points in the last neighborhood)

Mixed nested VND

- After exploring b (a parameter) neighborhoods, we switch from a nested to a sequential strategy. We can interrupt nesting at some level b ($1 \le b \le \ell_{max}$) 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

$$|\mathcal{N}_{\texttt{Mix-VND}}(x)| \le \sum_{\ell=b}^{\ell_{max}} |\mathcal{N}_\ell(x)| \times \prod_{\ell=1}^{b-1} |\mathcal{N}_\ell(x)|, \;\; x \in X.$$

n	Q	Local Search	Min. % dev	Max. % dev	Avg. % dev	Avg. time
200	10	forward-insertion	95.650	255.517	181.195	0.088
		backward-insertion	94.753	252.827	186.318	0.081
		2–opt	13.882	242.910	32.433	0.138
		Seq–VND	12.275	242.910	27.808	0.163
		Seq–VND–3	8.991	242.910	24.309	0.478
		Mix–VND	1.269	242.910	12.958	2.989
400	10	forward-insertion	78.320	218.770	165.603	0.385
		backward-insertion	81.881	218.770	169.416	0.317
		2–opt	12.078	217.104	21.852	0.831
		Seq–VND	10.684	217.104	18.954	0.769
		Seq–VND–3	8.951	203.738	16.035	4.062
		Mix–VND	1.492	217.104	6.573	26.569

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 `3L-VNS` $(x, k_{max}, t_{max}, \alpha)$
repeat

  $k \leftarrow 1$; $x_{best} \leftarrow x$
  repeat

    $x' \leftarrow$ `Shake`(x, k) /* Shaking */
    $x'' \leftarrow$ `VNDS`(x') /* VNDS Local search */
    if $f(x'') < f(x_{best})$ then $x_{best} \leftarrow x''$
    `NeighborhoodChangeS`$(x, x'', x_{best}, k, \alpha)$ /* Skewed move */
  until $k = k_{max}$;

  $t \leftarrow$ `CpuTime`$()$
until $t > t_{max}$;

- 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.

Formulation space search (FSS)

- Reformulation descent for Circle packing (Ml et al 2005)
- FSS for circle packing (Ml 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)

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 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

$$
\begin{aligned}
\min_{W,X} \quad & \sum_{i=1}^{n} \sum_{j=1}^{m} w_{ij} \|x_j - a_i\| \\
s.t. \quad \sum_{j=1}^{m} w_{ij} &= w_i, \quad i = 1, \ldots, n, \\
w_{ij} &\geq 0, \quad \forall i, j,
\end{aligned}
$$

- $a_i = (a_{i1}, a_{i2})$ 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, $i = 1, \ldots, n,\ j = 1, \ldots, m$;
- $\|x_j - a_i\| = [(x_{j1} - a_{i1})^2 + (x_{j2} - a_{i2})^2]^{1/2}$ 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
- Hansen, Mladenovic and Taillard (1998) p-median based.
- Brimberg *et al.* (2000) compared almost 20 different heuristics, including several new (GA, VNS, etc.
- Taillard (2002) Fixed neighborhood search based heuristic.
- Salhi, Gamal (2003) - GA
- Zainudin, Salhi (2007) perturbation based heuristic.
- Jabalameli, Ghaderi (2008) Hybrid Memetic and VNS.
- Brimberg, Mladenovic and Salhi (2008) survey.

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 $(n := n + card(U))$ 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 $n = |V|$.

	Objective function values			% deviation		CPU Times (sec.)	
m	Old best	VNS	FSS	VNS	FSS	VNS	FSS
5	1851879.88	1851879.88	1851877.25	0.00	0.00	3.78	69.50
10	1249564.75	1249564.75	1249564.75	0.00	0.00	5.87	27.11
15	980132.13	980146.25	980131.69	0.00	0.00	240.05	52.94
20	828802.00	828696.00	828685.69	-0.01	-0.01	32.28	27.92
25	722061.19	722053.56	722000.69	0.00	-0.01	140.16	82.83
30	638263.00	638236.00	638212.50	0.00	-0.01	175.02	135.98
35	577526.63	577523.94	577501.06	0.00	0.00	78.78	90.17
40	529866.19	529755.75	529680.81	-0.02	-0.03	289.02	86.83
45	489650.00	489551.00	489596.78	-0.02	-0.01	264.09	219.00
50	453164.00	453175.88	453149.09	0.00	0.00	137.34	271.00
55	422770.00	422903.06	422647.75	0.03	-0.03	293.13	168.95
60	397784.41	397759.50	397745.47	-0.01	-0.01	70.53	181.17
65	376759.50	376740.81	376630.47	0.00	-0.03	109.39	144.77
70	357385.00	357363.25	357335.34	-0.01	-0.01	277.00	133.50
75	340242.00	340208.19	340168.03	-0.01	-0.02	153.66	97.80
80	326053.19	326107.28	326035.28	0.02	-0.01	279.20	269.98

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

m	Objective function values Old best	VNS	FSS	% deviation VNS	FSS	CPU Times (sec.) VNS	FSS
85	313738.19	313769.16	313670.47	0.01	-0.02	289.56	70.36
90	302837.00	302816.50	302565.94	-0.01	-0.09	93.36	234.17
95	292875.09	292782.72	292730.59	-0.03	-0.05	228.70	44.75
100	283113.00	282964.00	282890.41	-0.05	-0.08	220.63	229.91
105	274576.00	274026.78	273705.91	-0.20	-0.32	83.75	298.28
110	265801.00	265329.50	265292.34	-0.18	-0.19	88.13	234.25
115	257605.00	257397.05	257243.98	-0.08	-0.14	200.45	241.50
120	249584.00	249532.33	249409.92	-0.02	-0.07	242.13	267.11
125	242930.00	242599.98	242111.36	-0.14	-0.34	297.59	296.81
130	236154.00	235945.00	235573.94	-0.09	-0.25	271.59	137.39
135	230431.00	229400.88	229169.05	-0.45	-0.55	272.52	154.27
140	224504.00	223530.78	223423.50	-0.43	-0.48	215.14	285.89
145	218279.00	218441.47	217791.58	0.07	-0.22	299.42	299.28
150	212926.00	212404.06	212344.36	-0.25	-0.27	155.97	156.73
Aver.	471575.19	471420.19	471315.09	-0.06	-0.10	183.61	171.88

Table 6: Comparison of VNS and FSS with $t_{max} = 300$ seconds, $k_{max} = p$ for both.

Variable formulation search

- 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 `Accept` (x, x', p)

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

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

	BVNS	**VFS$_1$**	**VFS$_2$**	**VFS$_3$**
Avg.	137.31	93.56	91.56	90.75
Dev. (%)	192.44	60.40	49.23	48.22
CPUt (s)	30.17	30.47	30.50	30.96

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.

HB (87)	**GPR**	**SA**	**SS**	**VFS**
Avg.	364.83	346.21	315.22	314.39
Dev. (%)	95.13	53.30	3.40	1.77
#Best	2	8	47	61
CPU t (s)	557.49	435.40	430.57	128.12

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