A continuous variable neighborhood search heuristic for finding the three-dimensional structure of a molecule

Milan Dražić a, Carlile Lavor b,*, Nelson Maculan c, Nenad Mladenović d

a Faculty of Mathematics, 11000 Belgrade, Serbia and Montenegro
b State University of Campinas (IMECC-UNICAMP), CP 6065, 13081-970, Campinas – SP, Brazil
c Federal University of Rio de Janeiro (COPPE-UFRJ), CP 68511, 21945-970, Rio de Janeiro – RJ, Brazil
d School of Mathematics, Brunel University, West London, Uxbridge UB8 3PH, United Kingdom

Received 1 June 2005; accepted 1 June 2006
Available online 18 October 2006

Abstract

We develop a continuous variable neighborhood search heuristic for minimizing the potential energy function of a molecule. Computing the global minimum of this function is very difficult because it has a large number of local minimizers which grows exponentially with molecule size. Experimental evidence shows that in the great majority of cases the global minimum potential energy of a given molecule corresponds to its three-dimensional structure and this structure is important because it dictates most of the properties of the molecule. Computational results for problems with up to 200 degrees of freedom are presented and favourable compared with other two existing methods from the literature. © 2006 Elsevier B.V. All rights reserved.

Keywords: Continuous variable neighborhood search; Molecular conformations; Global optimization

1. Introduction

The aim of this paper is to explore the capability of Variable Neighborhood Search (VNS for short) meta-heuristic to yield novel insights on the problem of finding the three-dimensional structure of a molecule. This structure is of particular importance because it is essential for understanding its functional mechanism and it is strongly related to the properties of the molecule [9].

The determination of the three-dimensional structure of a molecule can be formulated as a continuous global minimization problem. In the great majority of cases, that structure corresponds to the one involving the global minimum of the molecular potential energy function. The problem is that the number of local minimizers of this function grows exponentially with molecule size. Many optimization methods have been developed for this problem. They include simulated annealing, genetic algorithms, diffusion equation method, z BB algorithm, etc. For a survey, see [3,13,15].

* Corresponding author.
E-mail addresses: mdrazic@matf.bg.ac.yu (M. Dražić), clavor@ime.unicamp.br (C. Lavor), maculan@cos.ufrj.br (N. Maculan), Nenad.Mladenovic@brunel.ac.uk (N. Mladenović).

0377-2217/S - see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.ejor.2006.06.052
VNS is a metaheuristic usually used for solving combinatorial optimization problems. Few works deal with its application to the global minimization of functions depending on continuous variables [11,7], and in particular, to the minimization of molecular potential energy functions.

VNS metaheuristic developed in this paper uses different metrics in defining neighborhoods of the current best solution. To test the proposed methods and compare the results, we use a scalable simplified molecular potential energy function with well known properties established in [8]. Computational results for problems with up to 200 degrees of freedom are presented.

The outline of the paper is as follows. Section 2 presents the general principles of the VNS metaheuristic and its adaptation to the continuous case. Section 3 describes the molecular potential energy function that will be used to test the method. Section 4 gives the computational results and details of implementation in solving problem defined in Section 3. Finally, we end with some conclusions.

2. Continuous VNS metaheuristic

The variable neighborhood search (VNS) metaheuristic is well-established in the literature. For an overview of the method and numerous applications, the reader is referred to [4,5,10]. Basic steps of the VNS metaheuristic as seen in discrete optimization problems are given in Fig. 1.

The idea of VNS is to define a set of neighborhood structures \( \mathcal{N}_k \), \( k = 1, \ldots, k_{\text{max}} \), that can be used in a systematic way to conduct a search through the solution space. Whereas in local search for discrete problems a single neighborhood is typically defined (\( k_{\text{max}} = 1 \)), the VNS expands the search over an increasing radius to escape from a “local optimum trap”.

To induce a set of neighborhoods \( \mathcal{N}_k \) on the solution space \( S \), we use a distance function \( \rho \) that specifies the distance between any two points, \( x_1, x_2 \in S \). This may be done for example by comparing the attributes of the two solutions, and setting the distance equal to the number of attributes where \( x_1 \) and \( x_2 \) differ; that is, a Hamming distance is defined as

\[
q(x_1, x_2) = |x_1 \Delta x_2| = |(x_1 \setminus x_2) \cup (x_2 \setminus x_1)|.
\]

It is readily shown that \( \rho \) is a metric, and \((S, \rho)\) a metric space. For the continuous global optimization problem, where \( S \subseteq \mathbb{R}^n \), \( \rho(x_1, x_2) \) may be any metric, i.e., Euclidean, rectangular, \( l_p \) norm, etc.

The neighborhood \( \mathcal{N}_k(x) \) denotes the set of solutions in the \( k \)th neighborhood of \( x \), and using the metric \( \rho \), it is defined as balls

\[
\mathcal{N}_k(x) = \{ y \in S | \rho(x, y) \leq \rho_k \},
\]

or shells

\[
\mathcal{N}_k(x) = \{ y \in S | \rho_{k-1} \leq \rho(x, y) \leq \rho_k \},
\]

where \( \rho_k \) is the radius (size) of \( \mathcal{N}_k(x) \) monotonically increasing with \( k \).

---

 Initialization. Select the set of neighborhood structures \( \mathcal{N}_k \), \( k = 1, \ldots, k_{\text{max}} \), that will be used in the search; find an initial solution \( x \); choose a stopping condition;

 Repeat the following sequence until the stopping condition is met:

1. Set \( k \leftarrow 1 \);

2. Repeat the following steps until \( k = k_{\text{max}} \):

   a. **Shaking.** Generate a point \( y \) at random from the \( k \)-th neighborhood of \( x \) \(( y \in \mathcal{N}_k(x)) \);

   b. **Local search.** Apply some local search method with \( y \) as initial solution to obtain a local optimum \( y' \);

   c. **Move or not.** If this local optimum is better than the current best, move there \(( x \leftarrow y') \), and go to (1); otherwise, set \( k \leftarrow k + 1 \).

---

Fig. 1. Steps of the basic VNS.
The stopping condition may be, e.g., the predetermined maximal allowed CPU time ($t_{\text{max}}$), the maximal number of iterations, or the maximal number of iterations between two improvements. Let us note that the point $y$ is generated in Step 2(a) at random in order to avoid cycling which might occur if any deterministic rule was used.

In the recent work of Brimberg et al. [2], the global convergence of VNS is proved as well as its superiority over multistart local search (MLS) metaheuristic. In many continuous (as well as discrete) optimization problems, VNS performs better than genetic algorithm (GA) or tabu search (TS) metaheuristics [11,6,7].

Basic idea of VNS can be successfully applied to continuous global optimization problems:

$$\min_{x \in S} f(x), \quad S \subset \mathbb{R}^n.$$ 

In continuous optimization, contrary to discrete optimization, solution space and neighborhoods $N_k(x)$ are infinite sets. Therefore one cannot expect to fully explore any small neighborhood of a point in a local search, which is typical in discrete case. Nevertheless, we can apply some local minimization algorithm (steepest descent, Nelder–Mead, Newton, etc.) from starting point. Local minimum obtained by this minimizer can be far away from the starting point which we find to be a feature of the method because we are most of the time looking for a better solution lying in some distant part of a solution space.

For neighborhoods $N_k(x)$ in continuous case we can use $l_p$ metrics

$$\rho(x,y) = \left( \sum_{i=1}^{n} |x_i - y_i|^p \right)^{1/p}, \quad 1 < p < \infty,$$

$$\rho(x,y) = \max_{1 \leq i \leq n} |x_i - y_i|, \quad p = \infty.$$ 

These metrics lead to different geometric shapes of neighborhoods we explore.

In shaking step we generate a random point from $N_k(x)$ as a starting point for the local search. Distribution of this random point is another parameter of choice for the method. Uniform distribution is an obvious choice but some others can speed up the whole process.

Geometric neighborhood shapes and random point distributions used in the minimizing process can be changed after a number of unsuccessful steps to increase a chance for finding a better solution. Incorporating previous specific points for continuous optimization into the basic VNS framework we get continuous VNS metaheuristic with basic steps given in Fig. 2.

The continuous VNS metaheuristic does not have many parameters to specify so it appears to be a robust in that sense. Beside the stopping condition parameters (maximal allowed CPU time $t_{\text{max}}$, maximal number of

---

**Initialization.** Select the set of neighborhood structures $N_k$, $k = 1, \ldots, k_{\text{max}}$, that will be used in the search; choose a local minimizer; find an initial solution $x$; choose a stopping condition;

**Repeat** the following sequence until the stopping condition is met:

1. Set $k \leftarrow 1$;
2. **Repeat** the following steps until $k = k_{\text{max}}$:
   (a) **Shaking.** Generate a point $y$ at random from the $k$–th neighborhood of $x$ ($y \in N_k(x)$);
   (b) **Local search.** Apply local minimization method with $y$ as initial solution to obtain a local optimum $y'$;
   (c) **Move or not.** If this local optimum is better than the current best, move there ($x \leftarrow y'$), and go to (1); otherwise, set $k \leftarrow k + 1$.
3. **Reshape** Optionally change the set of neighborhood structures $N_k$ (geometry defined by metric) and random point distribution.

---

Fig. 2. Steps of the continuous VNS.
iterations, maximal number of iterations between two improvements), parameters that can influence the performance are:

- Number $k_{\text{max}}$ of neighborhood structures $\mathcal{N}_k$.
- Values of radii $\rho_i, i = 1, \ldots, k_{\text{max}}$. These values may be defined by user or calculated automatically in minimizing process.
- Geometry of neighborhood structures $\mathcal{N}_k$, defined by the choice of metric $\rho(x, y)$. Usual choices in continuous optimization are $l_1$, $l_2$, and $l_{\infty}$ metrics.
- Distribution used for obtaining the random point $y$ from $\mathcal{N}_k$ in shaking step. Uniform distribution in $\mathcal{N}_k$ is the obvious choice, but other distributions may lead to much better performance on some problems.
- Local optimizer used in local search step. A lot of local optimization algorithms are available both for smooth and not differentiable functions.

3. The problem

The potential energy of a molecule will be derived from molecular mechanics, which describes molecular interactions based on the principles of Newtonian physics. An empirically derived set of potential energy contributions is used for approximating these molecular interactions. This set of potential energy contributions, called the force field, contains adjustable parameters that are selected in such a way as to provide the best possible agreement with experimental data. Our discussion will focus on energy functions which share the main features of general molecular force fields.

The molecular model considered here consists of a chain of $N$ atoms centered at $x_1, \ldots, x_N \in \mathbb{R}^3$. For every pair of consecutive atoms $x_i$ and $x_{i+1}$, let $r_{ij}$ be the bond length which is the Euclidean distance between them. For every three consecutive atoms $x_i, x_{i+1}, x_{i+2}$, let $\theta_{ij}$ be the bond angle corresponding to the relative position of the third atom with respect to the line containing the previous two. Likewise, for every four consecutive atoms $x_i, x_{i+1}, x_{i+2}, x_{i+3}$, let $\omega_{ij}$ be the angle, called the torsion angle, between the normals through the planes determined by the atoms $x_i, x_{i+1,2}$ and $x_{i+1,3}$. The force field potentials corresponding to bond lengths, bond angles, and torsion angles will be defined respectively as

$$
E_1 = \sum_{(i,j) \in M_1} c_{ij}^1 (r_{ij} - \rho^0_{ij})^2,
$$

$$
E_2 = \sum_{(i,j) \in M_2} c_{ij}^2 (\theta_{ij} - \theta^0_{ij})^2,
$$

$$
E_3 = \sum_{(i,j) \in M_3} c_{ij}^3 (1 + \cos(3\omega_{ij} - \omega^0_{ij})),
$$

where $c_{ij}^1$ is the bond stretching force constant, $c_{ij}^2$ is the angle bending force constant, and $c_{ij}^3$ is the torsion force constant. The constants $\rho^0_{ij}$ and $\theta^0_{ij}$ represent the “preferred” bond length and bond angle, respectively, and the constant $\omega^0_{ij}$ is the phase angle that defines the position of the minima. The set of pairs of atoms separated by $k$ covalent bonds will be denoted by $M_k$ for $k = 1, 2, 3$.

In addition to the above, there is a potential $E_4$ which characterizes the 2-body interactions between every pair of atoms separated by more than two covalent bonds along the chain. We use the following function to represent $E_4$:

$$
E_4 = \sum_{(i,j) \in M_4} \left(\frac{(-1)^i}{r_{ij}}\right),
$$

where $r_{ij}$ is the Euclidean distance between atoms $x_i$ and $x_j$.

The general problem is the minimization of the total molecular potential energy function, $E_1 + E_2 + E_3 + E_4$, leading to the optimal spatial positions of the atoms. To reduce the number of parameters involved in the potentials above, we will simplify the problem considering a chain of carbon atoms. In this case, it is known that the preferred bond lengths are $\rho^0_{ij} = 1.526$ Å (for all $(i,j) \in M_1$) and that the bond angles
are \( \theta_{ij}^0 = 1.91 \text{ rad} \) (for all \((i,j) \in M_2\)). We will consider also that \( c_{ij}^1 = 1 \) (for all \((i,j) \in M_1\)), \( c_{ij}^2 = 1 \) (for all \((i,j) \in M_2\)), \( c_{ij}^3 = 1 \) (for all \((i,j) \in M_3\)), and \( \omega_{ij}^0 = 0 \) (for all \((i,j) \in M_3\)). While this structure reflects great simplifications over the general problem, its complexity should not be underestimated, as we will see below.

In most molecular conformational predictions, all covalent bond lengths and covalent bond angles are assumed to be fixed at their equilibrium values \( r_{ij}^0 \) and \( \theta_{ij}^0 \), respectively. Thus, the molecular potential energy function reduces to \( E_3 + E_4 \) and the first three atoms in the chain can be fixed. The first atom, \( x_1 \), is fixed at the origin, \((0,0,0)\); the second atom, \( x_2 \), is positioned at \((-r_{12},0,0)\); and the third atom, \( x_3 \), is fixed at \((r_{23}\cos(\theta_{13}) - r_{12},r_{23}\sin(\theta_{13}),0)\).

Using the parameters previously defined and Eqs. (1) and (2), we obtain

\[
E = \sum_{(i,j) \in M_3} (1 + \cos(3\omega_{ij})) + \sum_{(i,j) \in M_3} \left( \frac{(-1)^j}{r_{ij}} \right).
\]  

Although the molecular potential energy function (3) does not actually model the real system, it allows one to understand the qualitative origin of the large number of local minimizers – the main computational difficulty of the problem [15] – and is likely to be realistic in this respect.

Note that \( E_3 \), Eq. (1), is expressed as a function of torsion angles and \( E_4 \), Eq. (2), is expressed as a function of Euclidean distances. To represent (3) as a function of torsion angles only, we can use the result established in [14, p. 278] and obtain

\[
r_{ij}^2 = r_{ij}^2 + r_{ji}^2 - r_{ij} \left( \frac{r_{ji}^2 + r_{jk}^2 - r_{il}^2}{r_{jk}} \right) \cos(\theta_{ik}) - r_{ij} \left( \frac{\sqrt{4r_{ij}^2r_{jk}^2 - (r_{ij}^2 + r_{jk}^2 - r_{il}^2)^2}}{r_{jk}} \right) \sin(\theta_{ik}) \cos(\omega_{ij}),
\]

for every four consecutive atoms \( x_i, x_j, x_k, x_l \). Using the parameters previously defined, we have

\[
r_{ij} = \sqrt{10.60099896 - 4.141720682(\cos(\omega_{ij}))} \quad \text{for all} \ (i,j) \in M_3.
\]

From (3) and (4), the expression for the potential energy as a function of the torsion angles takes the form

\[
E = \sum_{(i,j) \in M_3} \left( 1 + \cos(3\omega_{ij}) + \frac{(-1)^j}{\sqrt{10.60099896,896 - 4.1411720682(\cos(\omega_{ij})}}} \right),
\]  

where \( i = 1, \ldots, N - 3 \) and \( N \) is the number of atoms in the given system.

In [8], it is shown that the number of local minimizers of the function (5) is \( 2^{N-3} \), where \( N \) is the number of atoms in the given system. Moreover, by imposing \( \omega_{ij} \in [0,5] \) for all \((i,j) \in M_3\), the existence of only one global minimizer is guaranteed. Independent of the number of variables, the global minimizer is the alternate sequence of torsion angles given by \( a,b,a,b,a,b,\ldots \), where \( a = 1.039195303 \) and \( b = 3.141592654 \), considering up to 10 digits.

The problem is then to find \( \omega_{14}, \omega_{25}, \ldots, \omega_{(N-3)N} \), considering \( \omega_{ij} \in [0,5] \), which correspond to the global minimum of the function \( E \), Eq. (5). \( E \) is a nonconvex function involving numerous local minimizers even for small molecules. These local minimizers “correspond” to metastable states of the molecule chain and the single global minimizer defines the energetically most favorable molecular conformation.

Despite these simplifications, the problem remains very difficult. A molecule with as few as 30 atoms has \( 2^{27} = 134,217,728 \) local minimizers. It can clearly be seen that finding the global minimum for chains of even moderate length is intractable via exhaustive methods [8].

4. VNS for molecular potential energy function

For finding the global minimum of the molecular potential energy function \( E \), Eq. (5), we developed a VNS based method for finding a global minimum:

\[
\text{global min} f(x),
\]

in a hyperrectangle \( X = \{(x_1,x_2,\ldots,x_n): a_i \leq x_i \leq b_i\} \).
According to the previous section, we defined the function $f(x)$ and box constraints for the solution domain as

$$f(x) = \sum_{i=1}^{n} \left( 1 + \cos(3x_i) + \frac{(-1)^i}{\sqrt{10.60099896 - 4.141720682 \cos(x_i)}} \right)$$

and

$$0 \leq x_i \leq 5, \quad i = 1, \ldots, n.$$

Three VNS heuristics are designed. They differ in the choice of random distributions used in the shaking step. For neighborhood structures $N_k, k = 1, \ldots, k_{\text{max}}$ we use shells $N_k(x) = \{ y \in X : R_{k-1} \leq \|y - x\|_{\infty} \leq R_k \}$. Radii $\rho_1 \leq \rho_2 \leq \cdots \leq \rho_{k_{\text{max}}}$ are automatically computed so that the ball with the largest radius covers the whole region $X$. In all three heuristics a random point is generated in two steps: first, a random direction is obtained, and after that, the random radius is determined in order to get a point in $N_k$.

(i) $VNS-1$. In the first heuristic a random direction is uniformly distributed in a unit $\ell_{\infty}$ ball. Random radius is chosen in such a way that the generated point is uniformly distributed in $N_k$.

(ii) $VNS-2$. In the second heuristic a random direction is determined by a random point uniformly distributed on a $\ell_1$ sphere.

(iii) $VNS-3$. In the third VNS based heuristic a random direction $x = (x_1, x_2, \ldots, x_n)$ is determined by a specially designed hypergeometric random point distribution on a unit $\ell_1$ sphere as follows:

(a) $x_1$ is taken uniformly on $[-1,1]$, $x_k$ is taken uniformly from $[-A_k, A_k]$, where $A_k = 1 - |x_{k-1}| - \cdots - |x_1|$, $k = 2, \ldots, n - 1$, and the last $x_n$ takes $A_n$ with random sign.

(b) Coordinates of $x$ are randomly permuted.

In order to diversify the search, we also implemented VNS heuristics that combine any two or all three basic heuristics. Namely, if no improvement in $k_{\text{max}}$ neighborhoods of the current structure is made, then we automatically (cyclically) change their structures for following iterations. $VNS-123$ denotes the VNS heuristic with all three previously described heuristics $VNS-1$, $VNS-2$, and $VNS-3$, used in that order.

### 4.1. Computer results

The results of the experiments for $n = 50, 100, 150, 200$ with $VNS-1$, $VNS-2$, $VNS-3$, and $VNS-123$ in ten runs are summarized in Table 1. Experiments presented in Tables 1 and 2 were performed on AMD 2500+ based PC platform with 512 MB of RAM. In Table 1, time limit was the stopping criteria and its values were chosen to document the influence of $k_{\text{max}}$ and various heuristics on overall performance. For each $k_{\text{max}}$, average and best % error of the function value from $f_{\text{best}}$ in ten runs are calculated as well as standard deviation of the % error, where % error of objective function value $f$ from $f_{\text{best}}$ is defined as $(f - f_{\text{best}})/f_{\text{best}} \times 100$. In all cases the local minimizer was steepest descent method with quadratic approximation method for one-dimensional optimization. Column $f_{\text{best}}$ contains exact values of global minima.

It can be seen from Table 1 that the basic VNS parameter $k_{\text{max}}$ has a strong influence on the quality of final results: the increase of $k_{\text{max}}$ leads to better performance. This fact was also confirmed with other test functions (see [7,12]). In cases with $k_{\text{max}} = 10$, $VNS-1$, $n = 150$ and $n = 200$, a local minimum (with $x_i = \pi$, $f = 0$) is quickly reached in all runs and better one was not found until $t_{\text{max}}$ (it is found, however, after that time limit).

It also appears that heuristic $VNS-3$ gives exceptionally good results, both alone or combined with other two. Namely, in all cases it reaches the optimal values. Due to the large values of $n$, the special distribution in $VNS-3$ generates random directions with only several coordinates significantly different from 0. Therefore, that special designed VNS heuristic for this class of problems pays off.

In order to confirm this last statement, more detailed results of $VNS-123$ and $VNS-3$, with $k_{\text{max}} = 15$, are reported in Table 2. In the second column of Table 2, the exact minimum of the function is given. Next columns present values obtained until that exact optimum is reached, averaged from ten repeated experiments:
the number of function and gradient calls ("fun" and "grad", respectively) as well as average running time in seconds ("sec") with the corresponding standard deviations ("dev").

From Table 2, it clearly appears that VNS-3 found the optimal solutions with much less efforts than VNS-123. Also, the important conclusion is that there is no exponential explosion of computer efforts with the increase of the problem size.

The same problem had been treated by using genetic algorithms [1] and branch and bound approach [8]. We summarize in Tables 3 and 4 some computational results obtained in [1] and [8], respectively.

In Table 3, for each problem size "n", "fmin" ("fmax") is the minimum (maximum) number of function evaluations needed to reach 99% of the global minimum in the 30 runs performed. Additionally, "favg" is the average and "fsd" is the standard deviation computed considering the successful runs ("succ"). The

### Table 1

<table>
<thead>
<tr>
<th>n</th>
<th>t_{max} (seconds)</th>
<th>Heuristic</th>
<th>k_{max} = 1</th>
<th>k_{max} = 5</th>
<th>k_{max} = 10</th>
<th>k_{max} = 15</th>
<th>J_{best}</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Aver (%)</td>
<td>Best (%)</td>
<td>Dev (%)</td>
<td>Aver (%)</td>
<td>Best (%)</td>
</tr>
<tr>
<td>50</td>
<td>5</td>
<td>VNS-1</td>
<td>82.50</td>
<td>70.64</td>
<td>7.25</td>
<td>80.73</td>
<td>51.86</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VNS-2</td>
<td>23.09</td>
<td>11.97</td>
<td>5.86</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VNS-3</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VNS-123</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>100</td>
<td>10</td>
<td>VNS-1</td>
<td>115.50</td>
<td>104.49</td>
<td>6.23</td>
<td>95.76</td>
<td>89.99</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VNS-2</td>
<td>70.40</td>
<td>55.21</td>
<td>9.68</td>
<td>4.98</td>
<td>1.99</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VNS-3</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VNS-123</td>
<td>0.20</td>
<td>0.00</td>
<td>0.60</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>150</td>
<td>15</td>
<td>VNS-1</td>
<td>130.92</td>
<td>124.76</td>
<td>4.29</td>
<td>96.22</td>
<td>89.29</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VNS-2</td>
<td>97.78</td>
<td>85.24</td>
<td>6.41</td>
<td>14.87</td>
<td>9.28</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VNS-3</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VNS-123</td>
<td>1.32</td>
<td>0.84</td>
<td>0.00</td>
<td>0.00</td>
<td>1.98</td>
</tr>
<tr>
<td>200</td>
<td>20</td>
<td>VNS-1</td>
<td>138.60</td>
<td>130.12</td>
<td>4.17</td>
<td>95.95</td>
<td>90.96</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VNS-2</td>
<td>103.90</td>
<td>95.75</td>
<td>5.39</td>
<td>28.01</td>
<td>19.96</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VNS-3</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VNS-123</td>
<td>6.17</td>
<td>1.99</td>
<td>1.93</td>
<td>5.78</td>
<td>2.99</td>
</tr>
</tbody>
</table>

### Table 2

<table>
<thead>
<tr>
<th>n</th>
<th>Minimum</th>
<th>VNS-123</th>
<th></th>
<th>VNS-3</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Fun</td>
<td>Dev</td>
<td>Grad</td>
<td>Dev</td>
<td>Sec</td>
</tr>
<tr>
<td>20</td>
<td>-0.82237</td>
<td>23,381</td>
<td>6711</td>
<td>1709</td>
<td>491</td>
</tr>
<tr>
<td>30</td>
<td>-1.23355</td>
<td>48,237</td>
<td>8242</td>
<td>3612</td>
<td>624</td>
</tr>
<tr>
<td>40</td>
<td>-1.64473</td>
<td>57,681</td>
<td>15,692</td>
<td>4414</td>
<td>1244</td>
</tr>
<tr>
<td>50</td>
<td>-2.05592</td>
<td>102,104</td>
<td>42,867</td>
<td>8009</td>
<td>3344</td>
</tr>
<tr>
<td>60</td>
<td>-2.46710</td>
<td>142,882</td>
<td>37,268</td>
<td>11,374</td>
<td>2983</td>
</tr>
<tr>
<td>70</td>
<td>-2.87828</td>
<td>172,091</td>
<td>68,178</td>
<td>13,979</td>
<td>5448</td>
</tr>
<tr>
<td>80</td>
<td>-3.28946</td>
<td>180,999</td>
<td>70,386</td>
<td>14,891</td>
<td>5724</td>
</tr>
<tr>
<td>90</td>
<td>-3.70065</td>
<td>211,160</td>
<td>46,778</td>
<td>17,766</td>
<td>3829</td>
</tr>
<tr>
<td>100</td>
<td>-4.11183</td>
<td>254,899</td>
<td>96,691</td>
<td>21,724</td>
<td>8230</td>
</tr>
<tr>
<td>120</td>
<td>-4.93420</td>
<td>375,970</td>
<td>103,186</td>
<td>33,035</td>
<td>9063</td>
</tr>
<tr>
<td>140</td>
<td>-5.75656</td>
<td>460,519</td>
<td>132,113</td>
<td>41,402</td>
<td>11,741</td>
</tr>
<tr>
<td>180</td>
<td>-7.40129</td>
<td>663,722</td>
<td>176,733</td>
<td>62,030</td>
<td>16,508</td>
</tr>
<tr>
<td>200</td>
<td>-8.22366</td>
<td>792,537</td>
<td>218,568</td>
<td>75,411</td>
<td>20,637</td>
</tr>
</tbody>
</table>
CPU time “sec” (in seconds) is also given. The program was coded in FORTRAN 77 and executed on a Pentium III 996 MHz with 192 MBytes of RAM memory.

In Table 4, again for each problem size “n”, “fun” is the number of function evaluations needed to reach the global minimum and “sec” is the CPU time in seconds (note the exponential increasing of the number of function evaluations with \( n \) and the time required to reach the global minimum for \( n = 25 \): 82.5 hours!). The program was coded in FORTRAN 90 and executed on a Pentium III 700 MHz with 256 MBytes of RAM memory.

From Tables 2–4, we can observe that all VNS versions perform much less computations than both genetic algorithm (GA) and branch and bound (B&B). Compare, e.g. for \( n = 20 \), the number of function evaluations obtained by VNS-3, GA and B&B, respectively: 10,569 (9887 + 682), 36,626, and 174,857. Also, because GA and B&B have bigger calculation overhead and use more memory than VNS, for the same time VNS can execute more function calls on the same computer. Note also that the codes for GA and B&B were run on slower computers.

5. Conclusions and research in progress

We developed VNS–based heuristics for minimization of a continuous function subject to box constraints. Moreover, for solving the molecular potential energy function, whose number of local minimizers exponentially grows with problem size, we suggest special VNS variant that appears to be very efficient. Results reported allow us to say that the ideas initially developed for discrete optimization successfully perform in case of continuous optimization problems.

Compared with the branch and bound scheme used for the same problem in [8] (see Tables 2 and 4), our VNS heuristics performed much better. Unlike exponential complexity of the first, our VNS based heuristics have approximately quadratic complexity on this minimization problem, as can be seen from the Table 2. Compared with genetic algorithms also used for the same problem in [1], VNS-123 and, particularly VNS-3, produced better results (see Tables 2 and 3).

Although VNS has only a few tuning parameters, their choice can increase the performance. Numerical experiments verify the conjecture that adding more neighborhood structures improves performance of the search. The choice of random point distribution in shaking step can also speed up the convergence significantly.

The future work will be focused on two directions: (i) extension of VNS to global optimization problems subject to general nonlinear constraints, and (ii) application of our VNS to more realistic molecular potential energy functions.
Acknowledgements

The authors would like to thank anonymous referees, whose comments greatly improved this paper, and for the support of CNPq.

References