Iranian Journal of Numerical Analysis and Optimization
Vol 5, No. 1, (2015), pp 13-36

A two-phase variable neighborhood search for solving nonlinear optimal control problems

R. Ghanbari*, A. Heydari and S. Nezhadhosein

Abstract

In this paper, a two-phase algorithm, namely IVNS, is proposed for solving nonlinear optimal control problems. In each phase of the algorithm, we use a variable neighborhood search (VNS), which performs a uniform distribution in the shaking step and the successive quadratic programming, as the local search step. In the first phase, VNS starts with a completely random initial solution of control input values. To increase the accuracy of the solution obtained from the phase 1, some new time nodes are added and the values of the new control inputs are estimated by spline interpolation. Next, in the second phase, VNS restarts by the solution constructed by the phase 1. The proposed algorithm is implemented on more than 20 well-known benchmarks and real world problems, then the results are compared with some recently proposed algorithms. The numerical results show that IVNS can find the best solution on 84% of test problems. Also, to compare the IVNS with a common VNS (when the number of time nodes is same in both phases), a computational study is done. This study shows that IVNS needs less computational time with respect to common VNS, when the quality of solutions are not different significantly.

Keywords: Nonlinear optimal control problem; Variable neighborhood search; Successive quadratic programming.

---

*Corresponding author
Received 19 April 2014; revised 29 December 2014; accepted 6 January 2014
R. Ghanbari
Department of Applied Mathematics, Faculty of Mathematical Science, Ferdowsi University of Mashhad, Mashhad, Iran. e-mail: rghanbari@um.ac.ir

A. Heydari
Department of Applied Mathematics, Payame Noor University, Mashhad, Iran. email: a_heidari@pnu.ac.ir

S. Nezhadhosein
Department of Applied Mathematics, Payame Noor University, Tehran, Iran. email: s_nejhadhosein@pnu.ac.ir

1 Introduction

Nonlinear optimal control problems (NOCP) are dynamic optimization problems with many applications in process systems engineering, including the design of trajectories for the optimal operation of batch and semi-batch reactors, economic systems, plasma physics, etc. [7].

Providing high-quality solutions with minimum computational time is the main issue for solving NOCPs. The numerical methods, direct [29] or indirect [46], usually have two main deficiencies, including low accuracy and convergence to a poor local solution. In direct methods, the quality of solutions depend on discretization resolution. These methods use control parametrization to convert continuous problems to discrete problems, so they may have less accuracy. However, the adaptive strategies [8, 43] can overcome these defects, but they may be trapped by a local optimal, yet. In the indirect approach, the problem using Pontryagins minimum principle (PMP) is converted to two boundary value problems (TBVP) and then it can be solved by numerical methods such as shooting method [29]. These methods need the good initial guesses that lie within the domain of convergence. Therefore, numerical methods are not usually suitable for solving NOCPs, especially for large-scale and multimodal models.

Metaheuristics as the global optimization methods can overcome these problems, but they usually need more computational time, though they don't really need good initial guesses and deterministic rules. Several researchers have used metaheuristics to solve optimal control problems. For instance, Michalewicz et al. [34] applied floating-point Genetic algorithms (GA) to solve discrete time optimal control problems, Yamashita and Shima [52] used the classical GAs to solve the free final time optimal control problems with terminal constraints. Abo-Hammour et al. [1] used continuous GA for solving NOCPs. Recently, Sun et al. [47] proposed a hybrid improved GA, for solving NOCPs and applied it for chemical processes. Moreover, the other usages of GA for optimal control problems can be found in [44, 45]. Modares and Naghibi-Sistani [37], proposed a hybrid algorithm by integrating an improved Particle Swarm Optimization (PSO) with a successive quadratic programming (SQP), for solving NOCPs. Lopez-Cruz et al. [14], applied Differential Evolution (DE) algorithms for solving the multimodal optimal control problems. Recently, Ghosh et al. [22] developed an ecologically inspired optimization technique, called Invasive Weed Optimization (IWO), for solving optimal control problems. The other well-known metaheuristic algorithms which are used for solving NOCPs are Genetic Programming (GP) [30], PSO [3, 4], Ant Colony Optimization (ACO) [48] and DE [31, 50].

Based on the success of the metaheuristics for solving NOCPs mentioned above, we propose an algorithm that use a well-known metaheuristic namely VNS (variable neighbourhood search) to solve NOCPs. Also, achieving a global optimal solution for NOCPs is another motivation for us to use a VNS [35]. VNS is an intelligent and metaheuristic method for solving a set

of combinatorial optimization and global optimization problems which uses neighborhood changes and uniform distributions in search procedure. Unlike many other metaheuristics, it is simple and requires few parameters [32]. Mladenović et al. [36] proposed a general VNS for solving continuous optimization. Moreover, VNS was used for solving several optimization problem [25] such as mixed integer programming [26], vertex weighted $k-$cardinality tree problem [10], and scheduling problem [13].

In this paper, VNS uses a uniform distribution in the shaking step and the SQP [39], as the local search step (similar to [37]). SQP is an iterative algorithm for solving NLP, which uses gradient information. Furthermore, SQP is used for solving NOCPs alone [6, 18].

For performing VNS to solve an NOCP, the time interval is uniformly divided by using a constant number of time nodes. Next, in each of these time nodes, the control variable is approximated by a scalar matrix of control input values. Thus, an infinite dimensional NOCP is changed to a finite dimensional nonlinear programming (NLP). Now, we encounter two conflict situations: the quality of the global solution and the needed computational time. In other words, when the number of time nodes is increased then we expect the quality of the global solution to increase but we know that in this situation the computational time is increased dramatically. In other situation, we consider less number of time nodes to reduce the computational but we may find a poor local solution. To conquer these problems, IVNS, performs VNS in two phases. In the first phase of IVNS (exploration phase), to decrease the computational time and to find a promising solution in the search space, VNS uses a less number of time nodes. Next to increase the quality of the solution obtained from Phase 1, the number of time nodes is increased. Using the obtained solution in Phase 1, the values of the new control inputs are estimated by spline interpolation. Next, in the second phase of IVNS (exploitation phase), VNS uses the solution constructed by the above procedure, as an initial solution. A computational study in our numerical experiments shows that there is a significant difference between the computational time of IVNS and a common VNS, that uses all time nodes from the beginning.

The rest of the paper is organized as follow: in Section 2, NOCPs are briefly introduced. In Section 3, IVNS is described. In Section 4, we provide more than 20 NOCPs to examine the numerical behaviour of the proposed algorithm. Results are compared with some numerical and metaheuristic methods. A computational study is carried out in Section 5 to show the effect of the second phase. We conclude in Section 6.

2 Problem formulation

NOCPs are formulated as optimization problems by the performance index as the objective function and differentiate equations as constraints that called dynamic optimizations. There are several types of these problems e.g. tracking problem, terminal control problem and time minimization problem [29]. We consider nonlinear bounded continuous-time control problems in which a vector of control functions, u, is exerted over the planning horizon $[t_0, t_f]$. The particular problem considered is that of finding the control input vector $u(t) \in \mathbb{R}^m$ that minimizes the performance index:

$$min \quad J = \phi(x(t_f), t_f) + \int_{t_0}^{t_f} g(x(t), u(t), t)dt \tag{1}$$

subject to:

$$\dot{x}(t) = f(x(t), u(t), t), \tag{2}$$
$$c(x(t), u(t), t) = 0, \tag{3}$$
$$d(x(t), u(t), t) \leq 0, \tag{4}$$
$$\psi(x(t_f), t_f) = 0, \tag{5}$$
$$x(t_0) = x_0, \qquad t \in [t_0, t_f]. \tag{6}$$

where $x(t) \in \mathbb{R}^n$ denotes the state vector for the system and $x_0 \in \mathbb{R}^n$ is the initial state. The functions $f : \mathbb{R}^{n+m} \times \mathbb{R} \to \mathbb{R}^n$, $g : \mathbb{R}^{n+m} \times \mathbb{R} \to \mathbb{R}$, $c : \mathbb{R}^{n+m} \times \mathbb{R} \to \mathbb{R}^{n_c}$, $d : \mathbb{R}^{n+m} \times \mathbb{R} \to \mathbb{R}^{n_d}$, $\psi : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^{n_\psi}$ and $\phi : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}$ are assumed to be sufficiently smooth on appropriate open sets. The cost function (1) must be minimized subject to dynamic (2), control and state equality constraints (3), control and state inequality constraints (4), the initial condition (6) and the final state constraints (5).

3 Proposed algorithm

Here, we propose IVNS for solving NOCPs. Before providing a description of IVNS, we introduce VNS.

3.1 VNS algorithm

VNS where introduced by Mladenović and Hansen in 1997 [35] is a metaheuristic algorithm which uses neighborhood changes systemically idea, both in the descent to local minima and in the escape from valleys which contain

local minima. It explores distant neighborhoods of the current incumbent solution, and moves from there to a new one if and only if an improvement is necessary. Local search method is applied repeatedly to get in the neighborhood to local optima [36]. Here, the implemented VNS in each phase has the following steps:
Initialization: The time interval is divided into N_t-1 subintervals using time nodes $t_0,\ldots,t_{N_t-1}$ and then control input values are computed (or selected randomly) as control points. This can be done by the following stages:

1. Let $t_k = t_0 + kh$, where $h = \frac{t_f - t_0}{N_t - 1}, k = 0, 1, \ldots, N_t - 1$, be time nodes, where t_0 and t_f are the initial and final times, respectively.

2. The corresponding control input value at each time node, $t_k, k = 0,\ldots,N_t-1$ is an $m \times 1$ vector, $u_k = [u_1^{(k)},\ldots,u_m^{(k)}]^T$, having the following components:

$$u_i^{(k)} = u_{left,i} + (u_{right,i} - u_{left,i}).r_i, \quad i = 1, 2, \ldots, m \tag{7}$$

where r_i is a random number in $[0, 1]$ with uniform distribution and $u_{left},\ u_{right} \in \mathbb{R}^m$ are the lower and the upper bound vectors of control input values, which can be given by the problem's definition or the user (e.g. see the NOCPs No. 7 and 8 in Appendix, respectively). $u = [u_k]_{k=0}^{N_t-1}$ is called control input matrix.

Evaluation: The corresponding state matrix with the control input matrix, u, is an $n \times N_t$ matrix, $x = [x_k]_{k=0}^{N_t-1}$, where x_k, $k = 0, 1, \ldots, N_t-1$, is an $n \times 1$ vector as the $(k+1)$-th column of state matrix, and can approximately be computed by the forth Runge-Kutta method on dynamic system (2) with the initial condition (6). Without loss of generality, assume $m = 1$ (for general case it can be extended easily). So, the evaluation procedure is as follows:

$$x_k = x_{k-1} + \frac{1}{6}(l_1 + 2l_2 + 2l_3 + l_4), \quad k = 1, 2, \ldots, N_t - 1 \tag{8}$$

where

$$l_1 = hf(x_k, u_k, t_k), \qquad l_2 = hf(x_k + \frac{l_1}{2}, u_k + \frac{h}{2}, t_k)$$
$$l_3 = hf(x_k + \frac{l_2}{2}, u_k + \frac{h}{2}, t_k), \qquad l_4 = hf(x_k + l_3, u_k + h, t_k)$$

where $u_k = u(t_k)$ and $x_k = x(t_k)$, with initial condition $x(t_0) = x_0$. To approximate the performance index, the composite Simpson's method [5], is used. Then, the performance index in (1), J, is approximated by $\tilde{J}$ as follows:

$$J \simeq \tilde{J} = \phi(x_{N_t-1}, t_{N_t-1}) + \frac{h}{3}(f_0 + 4\sum_{i=1}^{[\frac{N_t}{2}]-1} f_{2i+1} + 2\sum_{i=0}^{[\frac{N_t}{2}]-1} f_{2i} + f_{N_t-1}) \tag{9}$$

where $f_k = f(x_k, u_k, t_k)$, $k = 0, 1, \ldots N_t - 1$. If NOCP includes equality or inequality constraints e.g. (3) or (4), or has final state constraints, given by (5), then we add some penalties to the corresponding fitness value of the solution. Finally, we assign $I(u)$ to u as the fitness value as follows:

$$I(u) = \tilde{J} + \sum_{l=1}^{n_d} \sum_{j=0}^{N_t-1} M_{1l} max\{0, d_l(x_j, u_j, t_j)\} + \sum_{h=1}^{n_c} \sum_{j=0}^{N_t-1} M_{2h} c_h^2(x_j, u_j, t_j) + \sum_{p=1}^{n_\psi} M_{3p} \psi_p^2(x_{N_t-1}, t_{N_t-1}) \tag{10}$$

where $M_1 = [M_{11}, \ldots, M_{1n_d}]^T$, $M_2 = [M_{21}, \ldots, M_{2n_c}]^T$ and $M_3 = [M_{31}, \ldots, M_{3n_\psi}]^T$ are big numbers, as the penalty coefficients, for $c_h(.,.,.)$, $h = 1, 2, \ldots, n_c$, $d_l(.,.,.)$, $l = 1, 2, \ldots, n_d$, and $\psi_p(.,.)$, $p = 1, 2, \ldots, n_\psi$ defined in (3), (4) and (5), respectively.
The fitness value in (10), can be viewed as a nonlinear objective function with the decision variable as $u = [u_0, u_1, \ldots, u_{N_t-1}]$. This cost function with upper and lower bounds of input signals construct a finite dimensional NLP problem as follows:

$$\begin{aligned} &\min \quad I(u) = I(u_0, u_1, \ldots, u_{N_t-1}) \\ &s.t \\ &u_{left} \leq u_j \leq u_{right}, \quad j = 0, 1, \ldots, N_t - 1 \end{aligned} \tag{11}$$

Neighborhood: VNS uses at most k_{max} neighborhoods, $V_{r_1}, \ldots, V_{r_{k_{max}}}$, in which $r_i, i = 1, \ldots, k_{max}$ is the radii of i-th neighborhood, V_i, of the control input matrix u.
Shaking: In this stage, using a uniform distribution, a random direction matrix $d \in [-1, 1]^{m \times N_t}$ is firstly generated and then a random solution, $\bar{u}$, is selected in the k-th neighborhood, V_k, by the following equation:

$$\bar{u} = u + d.\alpha.(r + k - 1) \tag{12}$$

where $r \in [0, 1]$ is a random number, k is the index of neighborhood and α is the parameter of radii.
Local search: In this stage, SQP algorithm [9, 39] is performed on the NLP (11), using $\bar{u}^0 = \bar{u}$, constructed in (12), as the initial solution when the maximum number of iteration is *sqpmaxiter*.

SQP, is an effective and iterative algorithm for the numerical solution of the constrained NLP problem. This technique is based on finding a solution to the system of nonlinear equations that arise from the first-order necessary conditions for an extremum of the NLP problem. Using an initial solution of NLP, $\bar{u}^k$, $k = 0, 1, \ldots$, a sequence of solutions as $\bar{u}^{k+1} = \bar{u}^k + d^k$ is constructed, which d^k is the optimal solution of the constructed quadratic programming (QP) that approximates NLP in the iteration k based on $\bar{u}^k$,

as the search direction in the line search procedure. For the NLP (11), the principal idea is the formulation of a QP subproblem based on a quadratic approximation of the Lagrangian function as $L(u,\lambda) = I(u) + \lambda^T h(u)$, where the vector λ is Lagrangian multiplier and $h(u)$ return the vector of, inequality constraints evaluated at u. The QP is obtained by linearizing the nonlinear functions as follows:

$$\min \frac{1}{2} d^T H(\bar{u}^k) d + \nabla I(\bar{u}^k)^T d$$
$$\nabla h(\bar{u}^k)^T d + h(\bar{u}^k) \leq 0$$

Similar to [18], here a finite difference approximation is applied to compute the gradient of the cost function and the constraints, with the following components

$$\frac{\partial I}{\partial u_j} = \frac{I(...u_j + \delta...) - I(u_j)}{\delta}, \quad j = 0, 1, \ldots, N_t - 1 \tag{13}$$

where δ is the double precision of machine. So, the gradient vector is $\nabla I = [\frac{\partial I}{\partial u_0}, \ldots, \frac{\partial I}{\partial u_{N_t-1}}]^T$. Also, at each major iteration a positive definite quasi-Newton approximation of the Hessian of the Lagrangian function, H, is calculated using the BFGS method [39], where $\lambda_i,\ i = 1, ..., m$, is an estimated of the Lagrange multipliers. The general procedure of SQP, for NLP (11), is as follows:

1. Given an initial solution $\bar{u}^0$. Let $k = 0$.

2. Construct the QP subproblem (13), based on $\bar{u}^0$, using the approximations of the gradient and the Hessian of the the Lagrangian function.

3. Compute the new point as $\bar{u}^{k+1} = \bar{u}^k + d^k$, where d^k is the optimal solution of the current QP.

4. Let k=k+1 and go to step 2.

Here, in IVNS, SQP is used as the local search step, and we use the maximum number of iterations as the main criterion for stopping SQP. In other words, we terminate SQP when it converges either to local solution or the maximum number of SQP's iterations is reached.
Terminal conditions: The algorithm is terminated when the number of neighborhoods reached to k_{max} or the difference between cost functions in two consecutive iterations is less than ε (a given number).
VNS algorithm is given in Algorithm 1.

Algorithm 1 VNS algorithm

{Initialization} Input the number of time nodes N_t, the maximum number of iteration for SQP, $sqpmaxiter$, a maximum number of neighborhood, k_{max}, the parameter of radii, α defined in (12), the lower and the upper bound vectors of control input values u_{left}, u_{right}, an initial solution, u^*, and ε. Let $k = 1$.
{Evaluation} Evaluate the fitness of the initial solution, u^* and let $I^* = I(u^*)$, where $I(.)$ is defined in (10).
repeat
  {Shaking} Using (12), select u in k-th neighborhood of u^*.
  {Local search} Perform SQP algorithm on the NLP (11), using u as the initial solution when the maximum number of iteration is $sqpmaxiter$. Let $\bar{u}$ be the obtained solution, $\bar{I} = I(\bar{u})$ and $e = |\bar{I} - I^*|$.
  if $\bar{I} < I^*$ **then**
    Let $u^* = \bar{u}$, $I^* = \bar{I}$ and $k = 1$.
  else
    Let $k = k + 1$
  end if
until $k > k_{max}$ or $e < \varepsilon$
Return u^* as the approximate solution, x^* as the corresponding state and the corresponding fitness I^*.

3.2 IVNS

We now give a new algorithm, IVNS, which is a two-phase direct metaheuristic approach. The main idea of IVNS is to find promising solution of the search space using the computational time as few as possible.

IVNS has two main phases (as discussed in Section 1). In the first phase, we perform VNS (Algorithm 1) with a completely random initial solution constructed by (7). Since the main goal in this phase is to find the promising solution in the search space, we use a few number of time nodes.

Next, to maintain the property of the solution given in Phase 1 and to increase the accurately of this solution, we add some additional time nodes. Thus, we increase time nodes from N_{t_1} in the Phase 1 to N_{t_2} in the Phase 2. To use the information of the obtained solution from Phase 1 in the construction of the initial solution for Phase 2, we use Spline interpolation to estimate the values of the control inputs based on the curve obtained from the Phase 1. In the second phase, VNS restarts with this solution. Finally, IVNS is given in Algorithm 2.

Remark 3.1. *As we know, there are no general theorems on convergence of metaheuristics algorithm exist [28, 38]. Also, a specific theory on convergence of VNS does not exist, but a simple framework for global convergence of VNS based on attraction probabilities concept, can be found in [11]. However, we*

Algorithm 2 IVNS

Initialization Input u_{left} and u_{right}.
{**Phase 1**} Perform VNS (Algorithm 1) with a random initial solution and using the parameters N_{t_1}, *sqpmaxiter*, k_{max}, α and ε. (see Algorithm 1)
{**Constructing an initial solution for Phase 2**} Increase time nodes uniformly to N_{t_2} and estimate the corresponding control input values by using Spline interpolation on the obtained solution from Phase 1.
{**Phase 2**} Restart VNS (Algorithm 1) with the constructed initial solution and using N_{t_2}, *sqpmaxiter*, k_{max}, α and ε. (see Algorithm 1)

mentioned that all metaheuristics are practical algorithms that are interesting for their numerical behaviour, [16].

4 Numerical experiments

In this Section, to investigate the efficiency of IVNS, more than 20 well-known and real world NOCPs, as benchmark problems, are considered. These problems are selected with single control signal and multi control signals.

The numerical behaviour of the algorithms can be studied from two points of view: the performance index and the final state constraints. Let J be the value of the performance index and $\psi = [\psi_1, \ldots, \psi_{n_\psi}]^T$, defined in (5), and $\phi_f = \|\psi\|_2$ be the vector of final state constraints and the error of ψ, respectively. Now, the absolute errors for J and ϕ_f, are defined as follows:

$$E_J = |J - J^*|, \quad E_\psi = |\phi_f - \phi_f^*| \tag{14}$$

where J^* and $\phi_f^* = \|\psi^*\|_2$ are the best obtained solutions among the methods, or the exact solutions (when exist). To control the accuracy study, we now define a new criterion, called factor, to compare the numerical behaviour of the algorithms as follows:

$$K_\psi = E_J + E_\psi \tag{15}$$

where E_J and E_ψ are defined in (14). Note that K_ψ shows the summation of two important errors. Thus, based on K_ψ we can study the behaviour of algorithms on the quality and feasibility of given solutions, simultaneously.

To solve any NOCP described in the Appendix, we must know IVNS's parameters including N_{t_1}, N_{t_2}, k_{max}, α, ε and *sqpmaxiter* (see Algorithm 1), and the problem's parameters including u_{left}, u_{right} and M_i, $i = 1, 2, 3$, in (10). To estimate the best value of these parameters, for each problem, we run the proposed algorithm with different values of the parameters and then select the best. In all NOCPs, we consider the parameters *sqpmaxiter* $= 30$, $\alpha = 10^{-3}$ and $k_{max} = 10$. The other parameters are given in the associated

subsection or in Table 2. Because of the stochastic nature of the proposed algorithm, 12 different runs were done, for each NOCP, and the best result are reported in Table 1. The best value of each column is highlighted in the bold. The reported numerical results for each algorithm included the value of performance index, J, the absolute error of J and E_J, are defined in (14). The final state constraints, $\psi = [\psi_1, \ldots, \psi_{n_\psi}]^T$, the two-norm or error of the final state constraints, ϕ_f, the absolute error of ϕ_f and E_ψ, are defined in (14), and the factor K_ψ is defined in (15).

The algorithm was implemented in Matlab R2011a environment on a Notebook with Windows 7 Ultimate, CPU 2.53 GHz and 4.00 GB RAM. Also, to implement SQP in the proposed algorithm, as the local search, we used 'fmincon' in Matlab when the 'Algorithm' was set to 'SQP'.

In Subsection 4.1, the numerical results of IVNS are compared with exact solutions. Also, for comparing IVNS with metaheuristics and numerical algorithms in two Subsections 4.2 and 4.3, we consider 22 NOCPs. Their models are described in the Appendix, which are presented in terms of equations (1)-(6). The numerical results are summarized in Table 1. Details of these comparisons are given in the following subsection.

4.1 Comparison with the exact solution

Consider the nonlinear system state equations [24]

$$\dot{x}_1 = x_2^3,$$
$$\dot{x}_2 = u$$

The cost functional to be minimized, starting from the initial states $x_1(0) = 0$ and $x_2(0) = 1$, is

$$J = 4x_1(2) + x_2(2) + 4\int_0^2 u^2(t)dt$$

The exact trajectories of the problem, from PMP, are $x_1^*(t) = \frac{2}{5} - \frac{64}{5(t+2)^5}$ and $x_2^*(t) = \frac{4}{(t+2)^2}$, with the exact control signal $u^*(t) = \frac{-8}{(t+2)^3}$. Also the exact value of the performance index is $J^* = 3.35$. For the proposed algorithm, IVNS's parameters are set as $N_{t_1} = 15$, $N_{t_2} = 21$, $\varepsilon = 10^{-6}$ and the problem's parameters are set as $u_{left} = -1$ and $u_{right} = -\frac{1}{4}$. The IVNS's solution for the problem is $J = 3.3418$, thus, $E_J = K_\psi = 0.0082$.

Figure 1 shows the graphs of the exact and the obtained trajectories, for x_1 and x_2, and Figure 2 shows the graphs of the exact and the obtained control signals.

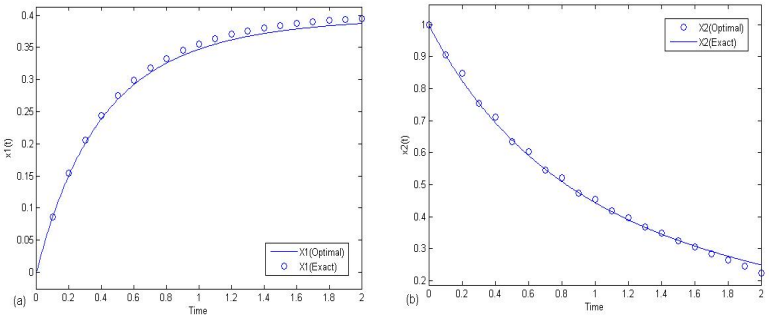

Figure 1: The exact and the obtained trajectories of (a) x_1 and (b) x_2, for the NOCP in subsection 4.1

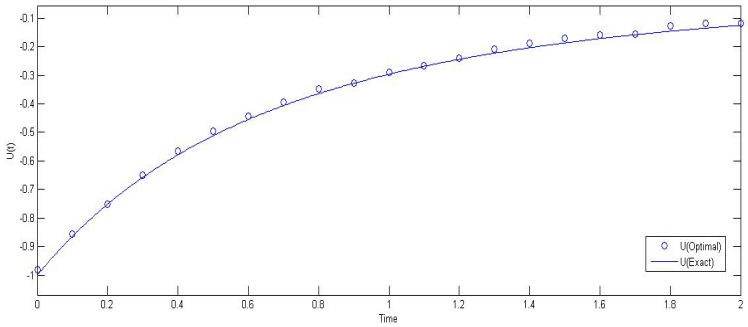

Figure 2: The exact and the obtained control signals for the NOCP in subsection 4.1

4.2 Comparison with metaheuristic algorithms

Here, six NOCPs are considered, NOCPs No. 1-6 in Appendix. The numerical results for the first NOCP is compared with hybrid improved GA, HIGA, proposed in [47]. The NOCPs No. 2-4, in the Appendix are compared with a metaheuristic, continuous GA and CGA, proposed in [1], which gave better solutions than shooting method and gradient algorithm (as the indirect methods) [29, 12], and SUMT (as the direct methods) [18]. For NOCPs No. 5 and 6 the results are compared with another metaheuristic, called IPSO, proposed in [37]. It has been shown that, for these NOCPs, IPSO was more accurate than some metaheuristic algorithms such as GA [42], DE [14], PSO [27] and some numerical methods [21, 23].

TCCR problem [47]

The first NOCP in the Appendix is a chemical process of Temperature Control for Consecutive Reaction, TCCR, which is an unconstrained two-state variable mathematical system. The objective is to obtain the optimal temperature profile that maximizes the yield of the temperature product B at the end of operation in a batch reactor, where the reaction $A \rightarrow B \rightarrow C$ is occurred. The state variables, x_1 and x_2 are the concentration of A and B, respectively, and the control variable u is the temperature. The problem solved by HIGA [47], which was more accurate than ACO [40] and iterative ACO [53]. From Table 1, we can see that the numerical behaviour of IVNS is better than HIGA.

VDP problem [1, 17]

The second NOCP in the Appendix is Van Der Pol, VDP, problem which has two state variables and one control variable. VDP problem has a final state constraint, which is $\psi = x_1(t_f) - x_2(t_f) + 1 = 0$. The problem solved by CGA [1] and IVNS. From [1], the norm of final state constraint for the CGA equals $\phi_f^* = 2.67 \times 10^{-11}$, however, this value for IVNS equals $\phi_f = 3.04 \times 10^{-9}$. So, the factor K_ψ for these methods can be seen in the sixth column of the Table 1. Note that the K_ψ of IVNS, 3.01×10^{-9}, is less than CGA's K_ψ, 3.0×10^{-4}. From Table 1, it is seem that IVNS can achieved more suitable solution than CGA.

CRP problem [1, 29]

The third NOCP in the Appendix is a mathematical model of Chemical Reactor Problem, CRP, which has two state variables and one control variable. The control variable is the flow of a coolant through a coil inserted in the reactor that controls the first-order irreversible exothermic reaction taking place in the reactor. The state variables, x_1 and x_2, are the deviations from the steady-state temperature and concentration, respectively. The numerical results of IVNS and CGA are shown in the third row of Table 1. CRP problem has two final state constraints, $\psi = [x_1, x_2]^T$. From [1], the norm of final state constraints for CGA, equals $\phi_f^* = 7.57 \times 10^{-10}$, when IVNS's norm of final state constraints is $\phi_f = 2.50 \times 10^{-8}$. But, the corresponding K_ψ of two methods shows that IVNS could achieve more accurate solutions than CGA.

FFRP problem [1, 18]

The fourth NOCP in the Appendix is Free Floating Robot Problem, FFRP, which has six state variables and four control variables. It was solved by CGA [1]. FFRP problem has six final state constraints, $\psi = [x_1 - 4, x_2, x_3 - 4, x_4, x_5, x_6]^T$. The norm of final state constraints for IVNS is $\phi_f^* = 4.61 \times 10^{-4}$, however, this value, from [1], for CGA is $\phi_f = 4.65 \times 10^{-3}$. From Table 1, we can see the numerical behaviour of IVNS is better than CGA, also it is clear that the obtained values of J, E_J, ϕ_f, E_ψ and K_ψ from IVNS are better than CGA.

CSTCR problem [37]

The fifth NOCP in the Appendix is a model of a nonlinear Continuous Stirred-tank Chemical Reactor, CSTCR. It has two state variables $x_1(t)$ and $x_2(t)$, as the deviation from the steady-state temperature and concentration, and one control variable $u(t)$, which represents the effect of the flow rate of cooling fluid on chemical reactor. The objective is to maintain the temperature and concentration close to steady-state values without expending large amount of control effort. Also, this is a benchmark problem in the handbook of test problems in local and global optimization [20], which is a multimodal optimal control problem [2]. It involves two different local minima. The values of the performance indices, for these solutions, equal 0.244 and 0.133. The numerical results of IVNS, with the parameters in Table 2, are compared with IPSO [37], and numerical methods in [2, 14]. From the results of the fifth row of Table 1, we can see that IVNS is the best.

MSNIC problem [37]

In the sixth NOCP in the Appendix, a Mathematical System with Nonlinear Inequality Constraint, MSNIC, is considered. It includes an inequality constraint, $d(x,t) = x_2(t) + 0.5 - 8(t - 0.5)^2 \leq 0$. From the sixth row of Table 1, we can see that the obtained value of the performance index, for IVNS is $J^* = 0.1720$, which is better than IPSO's, 0.1727, and other numerical methods given in [23, 33].

4.3 Comparison with numerical algorithms

In this subsection, for NOCPs no. 7-22, the results of IVNS are compared with some numerical methods such that SQP [18], SUMT [18], Bézier [21], HPM [15], DTM [41] and ADM [19]. Usually, for these methods the final

state constraints are not reported. But these values are reported for IVNS in Table 1.

Comparison with Bézier [21]

The NOCP No. 7, in the Appendix, has exact solution, i.e. the exact value of performance index equals $J^* = -5.5285$ [49]. This problem has an inequality constraint as $d(x,t) = -6 - x_1(t) \leq 0$. It has been solved by a numerical method, proposed in [21], called Bézier, and the proposed algorithm, IVNS, with the parameters in Table 2. From seventh row of Table 1, the obtained value of the performance index from IVNS is better and more accurate than Bézier method.

Comparison with HPM [15], DTM [41] and ADM [19]

In this subsection, the results of IVNS with the parameters given in Table 2, are compared with HPM [15], DTM [41] and ADM [19]. For NOCP No. 8 in the Appendix, which is a constraint nonlinear model, the numerical results are compared with HPM. This NOCP has a final state constraint as

$$\psi = x - 0.5 = 0.$$

From [15], the norm of final state constraint for HPM is $\phi_f = 4.2 \times 10^{-6}$, however, this value for IVNS equals $\phi_f^* = 6.83 \times 10^{-11}$. From Table 1, it is clear that the obtained values of the performance index, the norm of final state constraint and K_ψ from IVNS are better than HPM's.

The problem No. 9 in the Appendix is a linear quadratic optimal control which has been solved by two numerical methods, DTM [41] and ADM [19]. Using the approximate values of $k(t)$, which is used to achieve the optimal control signal by linear feedback control as $u(t) = -k(t)x(t)$, the performance index could be calculated. The exact solution, from PMP, equals $J^* = 0.1929$. From Table 1, the values of E_J and K_ψ, for IVNS, with the same number of points, $N_{t_2} = 15$, equals 0.0052, which is less than DTM and ADM methods, (0.0087).

Comparison with SQP and SUMT

For NOCPs No. 10-22 in the Appendix, the numerical results of IVNS (the parameters are given in Table 2) are compared with SQP and SUMT methods. All these problems are described in [18]. For SQP and SUMT, the status of the final state constraints were not reported, so, we replaced the values of ϕ_f instead of E_ψ, in Table 1. Also, in computation of the factor, K_ψ, the values of E_ψ for SQP and SUMT methods are considered to be zero. The

results (given in Table 1) show that IVNS could find more accurate results for performance index J, and the factor K_ψ, perspective.

Table 1: The best of numerical results for 12 different runs of NOCPs described in Appendix

Problem	Algorithm	J	E_J	E_ψ	K_ψ
TCCR	HIGA[47]	0.61046	2.0×10^{-5}	—	2.0×10^{-5}
	IVNS	**0.61048**	**0**	—	**0**
VDP	CGA [1]	1.7404	3.0×10^{-4}	**0**	3.0×10^{-4}
	IVNS	**1.7401**	**0**	3.01×10^{-9}	$\mathbf{3.01 \times 10^{-9}}$
CRP	CGA[1]	0.0163	4.0×10^{-4}	**0**	4.0×10^{-4}
	IVNS	**0.0159**	**0**	2.42×10^{-8}	$\mathbf{2.42 \times 10^{-8}}$
FFRP	CGA[1]	83.63	17.72	0.0042	17.7242
	IVNS	**65.91**	**0**	**0**	**0**
CSTCR	IPSO [37]	0.1354	0.0024	—	0.0024
	[2]	$J \in [0.135, 0.245]$	0.0020	—	0.0020
	[14]	$J \in [0.1358, 0.1449]$	0.0028	—	0.0028
	IVNS	**0.1328**	$\mathbf{2.0 \times 10^{-4}}$	—	$\mathbf{2.0 \times 10^{-4}}$
MSNIC	IPSO [37]	0.1727	0.0007	—	0.0007
	[23]	0.1816	0.0096	—	0.0096
	[33]	0.1769	0.0049	—	0.0049
	IVNS	**0.1720**	**0**	—	**0**
NOCP no. 7	Bézier [21]	-5.3898	0.1387	—	0.1387
	IVNS	$\mathbf{-5.5082}$	**0.0203**	—	**0.0203**
NOCP no. 8	HPM [15]	0.2353	0.0338	4.20×10^{-6}	0.0338
	IVNS	**0.2015**	**0**	**0**	**0**
NOCP no. 9	DTM [41]	0.2016	0.0087	—	0.0087
	ADM [19]	0.2016	0.0087	—	0.0087
	IVNS	**0.1877**	**0.0052**	—	**0.0052**
NOCP no. 10^b	SUMT [18]	5.15×10^{-6}	5.14×10^{-6}	—	5.14×10^{-6}
	SQP [18]	6.57×10^{-6}	6.56×10^{-6}	—	6.56×10^{-6}
	IVNS	$\mathbf{6.57 \times 10^{-11}}$	**0**	—	**0**
NOCP no. 11^b	SUMT [18]	1.7980	0.0791	—	0.0791
	SQP [18]	1.7950	0.0761	—	0.0761
	IVNS	**1.7189**	**0**	—	**0**
NOCP no. 12^b	SUMT [18]	0.1703	0.0223	—	0.0223
	SQP [18]	0.2163	0.0683	—	0.0683
	IVNS	**0.1480**	**0**	—	**0**
NOCP no. 13^b	SUMT [18]	3.2500	0.3507	NR[a]	0.3507
	SQP [18]	3.2500	0.3507	NR	0.3507
	IVNS	**2.8993**	**0**	7.49×10^{-10}	$\mathbf{7.49 \times 10^{-10}}$
NOCP no. 14^b	SUMT [18]	-0.2490	0.001	NR	0.001
	SQP [18]	-0.2490	0.001	NR	0.001
	IVNS	$\mathbf{-0.2500}$	**0**	2.6×10^{-10}	$\mathbf{2.6 \times 10^{-10}}$
NOCP no. 15^b	SUMT [18]	0.0167	6.0×10^{-4}	NR	6.0×10^{-4}
	SQP [18]	0.0168	7.0×10^{-4}	NR	7.0×10^{-4}
	IVNS	**0.0161**	**0**	3.42×10^{-9}	$\mathbf{3.42 \times 10^{-9}}$
NOCP no. 16^b	SUMT [18]	3.7700	0.4648	NR	0.4648
	SQP [18]	3.7220	0.4168	NR	0.4168
	IVNS	**3.3052**	**0**	3.35×10^{-8}	$\mathbf{3.35 \times 10^{-8}}$
NOCP no. 17^b	SUMT [18]	9.29×10^{-4}	3.0×10^{-6}	NR	3.0×10^{-6}
	SQP [18]	1.01×10^{-3}	8.4×10^{-5}	NR	8.4×10^{-5}
	IVNS	$\mathbf{9.26 \times 10^{-4}}$	**0**	6.66×10^{-10}	$\mathbf{6.66 \times 10^{-10}}$
NOCP no. 18^b	SUMT [18]	2.2080	0.2079	NR	0.2079
	SQP [18]	2.2120	0.2119	NR	0.2119
	IVNS	**2.0001**	**0**	5.01×10^{-11}	$\mathbf{5.01 \times 10^{-11}}$
NOCP no. 19^b	SUMT [18]	-8.8690	0.0002	NR	0.0002
	SQP [18]	-8.8690	0.0002	NR	0.0002
	IVNS	$\mathbf{-8.8692}$	**0**	6.89×10^{-9}	$\mathbf{6.89 \times 10^{-9}}$
NOCP no. 20^b	SUMT [18]	0.0368	0.0042	—	0.0042
	SQP [18]	0.0368	0.0042	—	0.0042
	IVNS	**0.0326**	**0**	—	**0**
NOCP no. 21^b	SUMT [18]	76.83	12.11	NR	12.11
	SQP [18]	77.52	12.80	NR	12.80
	IVNS	**64.72**	**0**	1.46×10^{-4}	$\mathbf{1.46 \times 10^{-4}}$
NOCP no. 22^b	SUMT [18]	0.3428	0.0670	NR	0.0670
	SQP [18]	0.3439	0.0681	NR	0.0681
	IVNS	**0.2758**	**0**	0.0021	**0.0021**

[a] Not Reported.
[b] We here consider, $E_\psi = \phi_f$ for IVNS, and for SQP and SUMT methods, $E_\psi = 0$ (since the values were not reported, we consider the best possible situation for SQP and SUMT).

Table 1 shows that IVNS was 100 percent successful in point of view the performance index, numerically. The associated values of E_J for IVNS are

zero for all test problems. It shows that IVNS provides robust results with respect to the other methods.

To have a more careful comparison, we computed the Gap between the performance index's value of the algorithms and the best obtained performance index's value. In other words, let J be the obtained value of the performance index of an algorithm. Now, similar to [51], we define the Gap as follows:

$$Gap(J) = |\frac{J - J^*}{J^*}| \tag{16}$$

From Table 1, the mean values of Gap for IVNS, SQP and SUMT, on NOCPs No. 10-22, are 0, $7.69e+3$ and $6.02e+3$, respectively. Thus it is obvious that, IVNS gave more better solution in comparison with SQP and SUMT. We believe that this is due to the fact that IVNS tries to find the global solution but SQP and SUMT didn't escape from a local minimum.

To compare with the CGA (as a global search algorithm), from Table 1, we see that the mean values of the Gap for CGA is 0.0981. Thus, we can see IVNS is 100 percent better than CGA from Gap perspective. This result shows that IVNS's estimations of global minimal is better than CGA's estimation. Therefore, based on these numerical study, we can conclude that IVNS outperforms than CGA.

The mean values of violation of the norm of the final state constraints, ϕ_f, for IVNS is 1.16×10^{-4}. Therefore, it is evident that IVNS is more robust. Also, the mean value of ϕ_f for IVNS and CGA are 1.53×10^{-4} and 1.55×10^{-3}, respectively, on NOCPs no. 2-4. Thus, we can say that the feasibility of the solutions given by IVNS and CGA are competitive. Therefore, it is seen that IVNS could provide very suitable solutions with respect to the optimality and feasibility criteria. Also, the mean of the factor, K_ψ, for IVNS equals 1.28×10^{-3}. For NOCPs No. 10-22 the mean of factor for IVNS, SQP and SUMT equals 1.76×10^{-4}, 1.0768 and 1.0272, respectively. Therefore, we can say that IVNS outperform well-known numerical methods. Since, the computational times of the most algorithms were not reported thus we didn't give the computational times of IVNS in Table 1. But, the details of the computational time of IVNS is given in Table 3 that will be discussed in Section 5.

5 Comparison with a common VNS

The main idea for proposing a two-phase algorithm is to decrease the required computational time in solving NOCPs. So, we focus on investigating of the influence of the second phase in IVNS. To compare the IVNS with a common VNS, the number of time nodes are selected same in both phases. In common VNS, only the first phase of IVNS, which the number of time node equal

Table 2: The parameters of IVNS for NOCPs described in the Appendix

Problem	u_{left}	u_{right}	N_{t_1}	N_{t_2}	ε	M_i
TCCR	298	398	11	15	10^{-6}	—
VDP	-0.5	2	31	151	10^{-6}	7
CRP	-1.5	2	21	51	10^{-8}	$[1,1]^T$
FFRP	-15	10	31	61	10^{-3}	$[70,\ldots,70]^T_{6\times1}$
CSTCR	0	5	31	51	10^{-9}	—
MSNIC	-20	20	21	51	10^{-3}	1
no. 7	-2	2	21	131	10^{-9}	1
no. 8	-2	2	31	91	10^{-6}	1
no. 9	-2	3	11	15	10^{-6}	—
no. 10	-3	3	21	51	10^{-6}	—
no. 11	-2	2	31	91	10^{-5}	1
no. 12	-20	20	31	51	10^{-8}	1
no. 13	-4	3	31	75	10^{-6}	$[100,100]^T$
no. 14	-1	1	31	71	10^{-6}	1000
no. 15	-2	2	21	41	10^{-6}	$[100,100]^T$
no. 16	$-\pi$	π	31	51	10^{-9}	$[100,100]^T$
no. 17	-1	1	21	35	10^{-6}	$[10,10]^T$
no. 18	-5	5	31	151	10^{-6}	$[10,10]^T$
no. 19	-30	30	31	171	10^{-6}	$[100,100]^T$
no. 20	-1	1	31	171	10^{-6}	—
no. 21	-15	10	31	71	10^{-6}	$[70,\ldots,70]^T_{6\times1}$
no. 22	-15	10	21	91	10^{-6}	$[10,\ldots,10]^T_{6\times1}$

N_{t_2}, is applied. For these methods, 35 different runs, for each NOCP in the Appendix, were made with the same parameters. The influence of these methods investigated for these NOCPs on the dependent outputs consist of performance index, J, the factor, ϕ_f and required computational time, $Time$. The results are given in Table 3.

From Table 3, we observe that the two-phase method has no significant effect on J, ϕ_f. But the two-phase method, IVNS, needs less computational time than the common VNS, significantly (except NOCP No. 16). Therefore, based on this computational study, we can conclude that the usage of two-phase VNS can decrease the computational time, significantly, without loss of quality of solution.

6 Conclusion

In this paper, a two-phase algorithm, namely IVNS, was proposed for solving NOCPs. In each phase of the algorithm, we used a VNS, which performed

Table 3: The best numerical results for NOCPs in Appendix, using IVNS and common VNS

	IVNS			VNS		
Problem	J	ϕ_f	$Time$	J	ϕ_f	$Time$
TCCR	0.6105	—	4.1496	0.6107	—	4.2482
VDP	1.7401	3.04×10^{-9}	375.69	1.7513	1.42×10^{-9}	413.28
CRP	0.0159	2.50×10^{-8}	78.09	0.0164	3.12×10^{-9}	112.05
FFRP	65.91	4.61×10^{-4}	264.62	50.31	8.17×10^{-3}	285.13
CSTCR	0.1328	—	48.82	0.1116	—	52.83
MSNIC	0.1720	—	10.49	0.1725	—	29.82
no. 7	-5.5082	—	42.27	-5.5012	—	65.81
no. 8	0.2015	6.83×10^{-11}	11.18	0.2012	4.21×10^{-10}	12.24
no. 9	0.1877	—	3.1278	0.1899	—	5.6636
no. 10	6.57×10^{-11}	—	3.7440	2.88×10^{-11}	—	3.9624
no. 11	1.7189	—	119.94	1.7152	—	139.55
no. 12	0.1480	—	41.38	0.1486	—	54.35
no. 13	2.8993	7.49×10^{-10}	39.04	2.8935	3.41×10^{-9}	38.36
no. 14	-0.2500	2.60×10^{-10}	52.61	-0.2498	1.52×10^{-8}	93.10
no. 15	0.0161	3.42×10^{-9}	124.02	0.0162	4.03×10^{-10}	154.65
no. 16	3.3052	3.35×10^{-8}	137.85	3.3051	1.03×10^{-10}	111.07
no. 17	9.26×10^{-4}	6.66×10^{-10}	144.16	9.81×10^{-4}	8.35×10^{-8}	178.23
no. 18	2.0001	5.01×10^{-11}	35.10	2.0001	2.13×10^{-12}	120.07
no. 19	-8.8692	6.89×10^{-9}	114.30	-8.8692	7.13×10^{-9}	129.02
no. 20	0.0326	—	42.69	0.0326	—	64.23
no. 21	64.72	1.46×10^{-4}	145.68	56.54	4.74×10^{-3}	148.01
no. 22	0.2758	0.0021	135.25	0.2765	0.0038	217.65

a uniform distribution in the shaking step and the SQP, as the local search step. In the first phase, VNS started with a completely random initial solution of control input values. To increase the accuracy of the solution obtained from Phase 1, the some new time nodes were added and the values of the new control inputs were estimated by Spline interpolation. Next, in the second phase, VNS restarted by the solution constructed by Phase 1. Finally, we implemented the proposed algorithm on more than 20 well-known benchmarks and real world problems, then the results were compared with some recently proposed algorithms. The numerical results showed that IVNS could found mostly better solution than other proposed algorithms. Also, to compare of IVNS with a common VNS a computational study was done that showed that IVNS needed less computational time with respect to a common VNS.

Acknowledgements

Authors are grateful to there anonymous referees and editor for their constructive comments.

References

1. Abo-Hammour, Z.S., Asasfeh, A.G., Al-Smadi, A.M. and Alsmadi, O.M.K. *A novel continuous genetic algorithm for the solution of optimal control problems,* Optimal Control Applications and Methods, 32(4) (2011) 414–432.

2. Ali, M.M., Storey, C. and Törn, A. *Application of stochastic global optimization algorithms to practical problems,* Journal of Optimization Theory and Applications, 95(3) (1997) 545–563.

3. Arumugam, M.S., Murthy, G.R. and Loo, C. K. *On the optimal control of the steel annealing processes as a two stage hybrid systems via PSO algorithms,* International Journal Bio-Inspired Computing, 1(3) (2009) 198–209.

4. Arumugam, M.S. and Rao, M.V.C. *On the improved performances of the particle swarm optimization algorithms with adaptive parameters, crossover operators and root mean square (RMS) variants for computing optimal control of a class of hybrid systems,* Application Soft Computing, 8(1) (2008) 324–336.

5. Atkinson, K. and Han, W. *Theoretical Numerical Analysis: A Functional Analysis Framework*, Texts in Applied Mathematics, Springer, 2009.

6. B*ü*skens, C. and Maurer, H. *SQP-methods for solving optimal control problems with control and state constraints: adjoint variables, sensitivity analysis and real-time control*, Journal of Computational and Applied Mathematics, 120(1-2) (2000) 85–108.

7. Betts, J.T. *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming*, Society for Industrial and Applied Mathematics, 2010.

8. Binder, T., Blank, L., Dahmen, W. and Marquardt, W. *Iterative algorithms for multiscale state estimation, part 1: Concepts*, Journal of Optimization Theory and Applications, 111(3) (2001) 501–527.

9. Bonnans, J.J.F., Gilbert, J.C., Lemaréchal, C. and Sagastizábal, C.A. *Numerical Optimization: Theoretical and Practical Aspects*, Springer London, Limited, 2006.

10. Brimberg, J., Uroevi, D. and Mladenović, N. *Variable neighborhood search for the vertex weighted k-cardinality tree problem*, European Journal of Operational Research, 171(1) (2006) 74 – 84.

11. Brimberg, J., Hansen, P. and Mladenović, N. *Attraction probabilities in variable neighborhood search*, 4OR, 8(2) (2010) 181–194.

12. Bryson, A.E. *Applied Optimal Control: Optimization, Estimation and Control*, Halsted Press book'. Taylor & Francis, 1975.

13. Costa, W., Goldbarg, M. and Goldbarg, E. *New VNS heuristic for total flowtime flowshop scheduling problem*, Expert Systems with Applications, 39(9) (2012) 8149–8161.

14. Lopez Cruz, I.L., Van Willigenburg, L.G. and Van Straten, G. *Efficient differential evolution algorithms for multimodal optimal control problems*, Applied Soft Computing, 3(2) (2003) 97–122.

15. Effati, S. and Saberi Nik, H. *Solving a class of linear and non-linear optimal control problems by homotopy perturbation method,* IMA Journal of Mathematical Control and Information, 28(4) (2011) 539–553.

16. Engelbrecht, A.P. *Computational Intelligence: An Introduction*, Wiley, 2007.

17. Fabien, B.C. *Numerical solution of constrained optimal control problems with parameters*, Applied Mathematics and Computation, 80(1) (1996) 43–62.

18. Fabien, B.C. *Some tools for the direct solution of optimal control problems*, Advances Engineering Software, 29(1) (1998) 45–61.

19. Fakharian, A., Beheshti, M.T.H. and Davari, A. *Solving the Hamilton -Jacobian-Bellman equation using adomian decomposition method*, International Journal of Computer Mathematics, 87(12) (2010) 2769–2785.

20. Floudas, C.A. and Pardalos, P.M. *Handbook of test problems in local and global optimization, Nonconvex optimization and its applications*, Kluwer Academic Publishers, 1999.

21. Ghomanjani, F., Farahi, M.H. and Gachpazan, M.*Bézier control points method to solve constrained quadratic optimal control of time varying linear systems*, Computational and Applied Mathematics, 31 (2012) 433–456.

22. Ghosh, A., Das, S., Chowdhury, A. and Giri, R. *An ecologically inspired direct search method for solving optimal control problems with Bézier parameterization*, Engineering Applications of Artificial Intelligence, 24(7) (2011) 1195–1203.

23. Goh, C.J. and Teo, K.L. *Control parametrization: A unified approach to optimal control problems with general constraints,* Automatica, 24(1) (1988) 13–18.

24. Gong, Q., Kang, W. and Ross, I.M. *A pseudospectral method for the optimal control of constrained feedback linearizable systems*, Automatic Control, IEEE Transactions on, 51(7) (2006) 1115–1129.

25. Hansen, P., Mladenović, N. and Pérez, J. M. *Variable neighbourhood search: methods and applications,* 4OR, 6(4) (2008) 319–360.

26. Hansen, P., Mladenović, N. and Urošević, D. *Variable neighborhood search and local branching,* Computers and Operations Research, 33(10) (2006) 3034–3045.

27. Herrera, F. and Zhang, J. *Optimal control of batch processes using particle swam optimisation with stacked neural network models,* Computers and Chemical Engineering, 33(10) (2009) 1593–1601.

28. Johnson, A.W. and Jacobson, S.H. *On the convergence of generalized hill climbing algorithms,* Discrete Applied Mathematics, 119(1-2) (2002) 37–57.

29. Kirk, D.E. *Optimal Control Theory: An Introduction*, Dover Publications, 2004.

30. Vincent Antony Kumar, A. and Balasubramaniam, P. *Optimal control for linear system using genetic programming,* Optimal Control Applications and Methods, 30(1) (2009) 47–60.

31. Lee, M.H., Han, C. and Chang, K.S. *Dynamic optimization of a continuous polymer reactor using a modified differential evolution algorithm,* Industrial and Engineering Chemistry Research, 38(12) (1999) 4825–4831.

32. Loudni, S., Boizumault, P. and Levasseur, N. *Advanced generic neighborhood heuristics for VNS*, Engineering Applications of Artificial Intelligence, 23(5) (2010) 736–764.

33. Mekarapiruk, W. and Luus, R. *Optimal control of inequality state constrained systems*, Industrial and Engineering Chemistry Research, 36(5) (1997) 1686–1694.

34. Michalewicz, Z., Janikow, C.Z. and Krawczyk, J.B. *A modified genetic algorithm for optimal control problems*, Computers & Mathematics with Applications, 23(12) (1992) 83–94.

35. Mladenović, N. and Hansen, P. *Variable neighborhood search*, Computers and Operations Research, 24(11) (1997) 1097–1100.

36. Mladenović, N., Dražić, M., Kovačevic-Vujčić, V. and Čangalović, M. *General variable neighborhood search for the continuous optimization*, European Journal of Operational Research, 191(3) (2008) 753–770.

37. Modares, H. and Naghibi-Sistani, M.B. *Solving nonlinear optimal control problems using a hybrid IPSO - SQP algorithm*, Engineering Applications of Artificial Intelligence, 24(3) (2011) 476–484.

38. Muhlenbein, H. and Zimmermann, J. *Size of neighborhood more important than temperature for stochastic local search*, In Evolutionary Computation, 2000. Proceedings of the 2000 Congress on, volume 2, pages 1017–1024 vol.2, 2000.

39. Nocedal, J. and Wright, S.J. *Numerical Optimization*, Springer series in operations research and financial engineering, Springer, 1999.

40. Rajesh, J., Gupta, K., Kusumakar, H.S., Jayaraman, V.K. and Kulkarni, B.D. *Dynamic optimization of chemical processes using ant colony framework*, Computers & Chemistry, 25(6) (2001) 583–595.

41. Saberi Nik, H., Effati, S. and Yildirim, A. *Solution of linear optimal control systems by differential transform method,* Neural Computing and Applications, 23(5) (2013) 1311–1317.

42. Sarkar, D. and Modak, J.M. *Optimization of fed-batch bioreactors using genetic algorithm: multiple control variables,* Computers and Chemical Engineering, 28(5) (2009) 789–798.

43. Schlegel, M., Stockmann, K., Binder, T. and Marquardt, W. *Dynamic optimization using adaptive control vector parameterization,* Computers & Chemical Engineering, 29(8) (2005) 1731–1751.

44. Shi, X.H., Wan, L.M., Lee, H.P., Yang, X.W., Wang, L.M. and Liang, Y.C. *An improved genetic algorithm with variable population-size and a PSO-GA based hybrid evolutionary algorithm,* Machine Learning and Cybernetics, 2003 International Conference on, volume 3, pages 1735–1740, 2003.

45. Sim, Y.C., Leng, S.B. and Subramaniam, V. *A combined genetic algorithms-shooting method approach to solving optimal control problems,* International Journal of Systems Science, 31(1) (2000) 83–89.

46. Srinivasan, B., Palanki, S. and Bonvin, D. *Dynamic optimization of batch processes: I. characterization of the nominal solution,* Computers & Chemical Engineering, 27(1) (2003) 1 – 26.

47. Sun, F., Du, W., Qi, R., Qian, F. and Zhong, W. *A hybrid improved genetic algorithm and its application in dynamic optimization problems of chemical processes,* Chinese Journal of Chemical Engineering, 21(2) (2013) 144–154.

48. Marinus van Ast, J., Babuška, R. and De Schutter, B. *Novel ant colony optimization approach to optimal control,* International Journal of Intelligent Computing and Cybernetics, 2(3) (2009) 414–434.

49. Vlassenbroeck, J. *A chebyshev polynomial method for optimal control with state constraints,* Automatica, 24(4) (1988) 499–506.

50. Wang, F.S. and Chiou, J.P. *Optimal control and optimal time location problems of differential-algebraic systems by differential evolution,* Industrial and Engineering Chemistry Research, 36(12) (1997) 5348–5357.

51. Yaghini, M., Karimi, M. and Rahbar, M. *A hybrid metaheuristic approach for the capacitated p-median problem,* Applied Soft Computing, 13(9) (2013) 3922–3930.

52. Yamashita, Y. and Shima, M. *Numerical computational method using genetic algorithm for the optimal control problem with terminal constraints and free parameters,* Nonlinear Analysis: Theory, Methods & Applications, 30(4) (1997) 2285–2290.

53. Zhang, B., Chen, D. and Zhao, W. *Iterative ant-colony algorithm and its application to dynamic optimization of chemical process,* Computers & Chemical Engineering, 29(10) (2005) 2078–2086.

Appendix

The following NOCPs are described using eqns (1)-(6).

1. [47, 53, 40] (TCCR) $\phi = x_2,\ t_0 = 0,\ t_f = 1,\ f = [-4000exp(-2500/u)x_1^2, 4000exp(-2500/u)x_1^2 - 620000exp(-5000/u)x_2]^T,\ d = [298-u, u-398]^T,\ x_0 = [1,0]^T$.

2. [1, 17] (VDP) $g = \frac{1}{2}(x_1^2 + x_2^2 + u^2), t_0 = 0, t_f = 5, f = [x_2, -x_2 + (1 - x_1^2)x_2 + u]^T, x_0 = [1,0]^T, \psi = x_1 - x_2 + 1$.

3. [1, 29] (CRP) $g = \frac{1}{2}(x_1^2+x_2^2+0.1u^2), t_0 = 0, t_f = 0.78, f = [x_1 - 2(x_1 + 0.25) + (x_2 + 0.5)exp(25x_1/(x_1 + 2)) - (x_1 + 0.25)u, 0.5 - x_2 - (x_2 + 0.5)exp(25x_1/(x_1 + 2))]^T, x_0 = [0.05, 0]^T, \psi = [x_1, x_2]^T$.

4. [1, 18] (FFRP) $g = \frac{1}{2}(u_1^2 + u_2^2 + u_3^2 + u_4^2), t_0 = 0, t_f = 5, f = [x_2, ((u_1 + u_2)\cos x_5 - (u_2 + u_4)\sin x_5)/M, x_4, ((u_1 + u_3)\sin x_5 + (u_2 + u_4)\cos x_5)/M, x_6, (D(u_1 + u_3) - L_e(u_2 + u_4))/I]^T, x_0 = [0,0,0,0,0,0]^T, \psi = [x_1 - 4, x_2, x_3 - 4, x_4, x_5, x_6]^T, M = 10, D = 5, I = 12, L_e = 5$.

5. [37] (CSTCR) $g = x_1^2 + x_2^2 + 0.1u^2, t_0 = 0, t_f = 0.78, f = [-(2 + u)(x_1 + 0.25) + (x_2 + 0.5)exp(25x_1/(x_1 + 2)), 0.5 - x_2 - (x_2 + 0.5)exp(25x_1/(x_1 + 2))]^T, x_0 = [0.09, 0.09]^T$.

6. [37] (MSNIC) $\phi = x_3, t_0 = 0, t_f = 1, f = [x_2, -x_2 + u, x_1^2 + x_2^2 + 0.005u^2]^T, d = [-(20-u)(20+u), x_2 + 0.5 - 8(t-0.5)^2]^T, x_0 = [0, -1, 0]^T$.

7. [21] $g = 2x_1, t_0 = 0, t_f = 3, f = [x_2, u]^T, d = [-(2 - u)(2 + u), -6 - x_1]^T, x_0 = [2, 0]^T$.

8. [15] $g = u^2, t_0 = 0, t_f = 1, f = \frac{1}{2}x^2 \sin x + u,\ x_0 = 0, \psi = x - 0.5.$

9. [41, 19] $g = \frac{1}{2}(x^2 + u^2),\ t_0 = 0, t_f = 1,\ f = -x + u,\ x_0 = 1.$

10. [18] $g = x^2 \cos^2 u,\ t_0 = 0, t_f = \pi,\ f = \sin \frac{u}{2},\ x_0 = \frac{\pi}{2}.$

11. [18] $g = \frac{1}{2}(x_1^2 + x_2^2 + u^2), t_0 = 0, t_f = 5, f = [x_2, -x_1 + (1 - x_1^2)x_2 + u]^T, d = -(x_2 + 0.25), x_0 = [1, 0]^T.$

12. [18] $g = x_1^2 + x_2^2 + 0.005u^2, t_0 = 0, t_f = 1, f = [x_2, -x_2 + u]^T, d = [-(20 - u)(20 + u), 0.5 + x_2 - 8(t - 0.5)^2]^T, x_0 = [0, -1]^T.$

13. [18] $g = \frac{1}{2}u^2, t_0 = 0, t_f = 2, f = [x_2, u]^T, x_0 = [1, 1]^T, \psi = [x_1, x_2]^T.$

14. [18] $g = -x_2, t_0 = 0, t_f = 1, f = [x_2, u]^T, d = -(1 - u)(1 + u), x_0 = [0, 0]^T, \psi = x_2.$

15. [18] $g = \frac{1}{2}(x_1^2 + x_2^2 + 0.1u^2), t_0 = 0, t_f = 0.78, f = [-2(x_1 + 0.25) + (x_2 + 0.5)exp(25x_1/(x_1 + 2)) - (x_1 + 0.25)u, 0.5 - x_2 - (x_2 + 0.5)exp(25x_1/(x_1 + 2))]^T, x_0 = [0.05, 0]^T, \psi = [x_1, x_2]^T.$

16. [18] $g = \frac{1}{2}u^2, t_0 = 0, t_f = 10, f = [\cos u - x_2, \sin u]^T, d = -(\pi - u)(\pi + u), x_0 = [3.66, -1.86]^T, \psi = [x_1, x_2]^T.$

17. [18] $g = \frac{1}{2}(x_1^2 + x_2^2), t_0 = 0, t_f = 0.78, f = [-2(x_1 + 0.25) + (x_2 + 0.5)exp(25x_1/(x_1 + 2)) - (x_1 + 0.25)u, 0.5 - x_2 - (x_2 + 0.5)exp(25x_1/(x_1 + 2))]^T, d = -(1 - u)(1 + u), x_0 = [0.05, 0]^T, \psi = [x_1, x_2]^T.$

18. [18] $\phi = x_3, t_0 = 0, t_f = 1, f = [x_2, u, \frac{1}{2}u^2]^T, d = x_1 - 1.9, x_0 = [0, 0, 0]^T, \psi = [x_1, x_2 + 1]^T.$

19. [18] $\phi = -x_3, t_0 = 0, t_f = 5, f = [x_2, -2 + \frac{u}{x_3}, -0.01u]^T, d = -(30 - u)(30 + u), x_0 = [10, -2, 10]^T, \psi = [x_1, x_2]^T.$

20. [18] $\phi = (x_1 - 1)^2 + x_2^2 + x_3^2,\ g = \frac{1}{2}u^2,\ t_0 = 0, t_f = 5, f = [x_3 \cos u, x_3 \sin u, \sin u]^T, x_0 = [0, 0, 0]^T.$

21. [18] $g = \frac{1}{2}(u_1^2 + u_2^2 + u_3^2 + u_4^2), t_0 = 0, t_f = 5, f = [x_2, ((u_1 + u_3)\cos x_5 - (u_2 + u_4)\sin x_5)/M, x_4, ((u_1 + u_3)\sin x_5 + (u_2 + u_4)\cos x_5)/M, x_6, (D(u_1 + u_3) - L_e(u_2 + u_4))/I]^T, x_0 = [0, 0, 0, 0, 0, 0]^T, \psi = [x_1 - 4, x_2, x_3 - 4, x_4, x_5 - \frac{\pi}{4}, x_6]^T, M = 10, D = 5, I = 12, L_e = 5.$

22. [18] $g = 4.5(x_3^2 + x_6^2) + 0.5(u_1^2 + u_2^2), t_0 = 0, t_f = 1, f = [9x_4, 9x_5, 9x_6, 9(u_1 + 17.25x_3), 9u_2, -9(u_1 - 27.0756x_3 + 2x_5x_6)/x_2]^T, x_0 = [0, 22, 0, 0, -1, 0]^T, \psi = [x_1 - 10, x_2 - 14, x_3, x_4 - 2.5, x_5, x_6]^T.$

یک الگوریتم دوفازی جستجوی همسایگی متغیر برای حل مسائل کنترل بهینه غیرخطی

رضا قنبری[۱]*، عقیله حیدری*[۲] *و سعید نژاد حسین*[۳]

[۱] دانشگاه فردوسی مشهد، دانشکده علوم ریاضی، گروه ریاضی کاربردی
[۲] دانشگاه پیام نور، مشهد، گروه ریاضی کاربردی
[۳] دانشگاه پیام نور، تهران، گروه ریاضی کاربردی

چکیده : در این مقاله یک الگوریتم دوفازی به نام IVNS برای حل مسائل کنترل بهینه غیرخطی پیشنهاد شده است. در هر فاز الگوریتم پیشنهادی از روش جستجوی همسایگی متغیر (VNS) استفاده می‌کنیم که در آن از توزیع یکنواخت در گام لغزشو از روش برنامه‌ریزی درجه دو دنباله‌ای در گام جستجوی محلی استفاده شده است. در فاز اول، الگوریتم VNS با یک جواب اولیه کاملا تصادفی از متغیرهای ورودی کنترل اجرا می‌شود. به منظور افزایش دقت جواب بدست آمده از فاز اول، نقاطگره‌ای زمانی جدیدی اضافه می‌شوند و مقادیر ورودی کنترل در آنها با درون‌یابی اسپلاین تقریب زده می‌شوند. سپس در فاز دوم VNS با جواب جدید ساخته شده از فاز اول مجددا راه اندازی می‌شود. الگوریتم پیشنهادی روی ۲۰ مساله کنترل بهینه واقعی، به عنوان مسائل آزمون، پیاده سازی شده است. نتایج عددی با برخی از الگوریتم‌های پیشنهادی اخیر مقایسه شده است. نتایج نشان می‌دهد روش پیشنهادی جواب‌های عددی بهتری نسبت به سایر روش‌ها در زمان محاسباتی کمتر ارائه می‌دهد. همچنین برای مقایسهIVNS با VNS تک فاز (که تعداد نقاط گره‌ای دردو فاز مشابه است) یک آزمایش عددی انجام شده است. این مطالعه نشان داد IVNS زمان محاسباتی کمتری نسبت به VNS دارد، در حالی که کیفیت جواب‌های بدست آمده به صورت معنی داری تغییر نمی‌کند.

کلمات کلیدی : مساله کنترل بهینه غیرخطی؛ جستجوی همسایگی متغیر؛ برنامه ریزی درجه دو دنباله ای.