Advances in Engineering Software 45 (2012) 11–20

Contents lists available at SciVerse ScienceDirect

Advances in Engineering Software

journal homepage: www.elsevier.com/locate/advengsoft

Approximate branch-and-bound global optimization using B-spline hypervolumes

Sangkun Park *

Department of Mechanical Engineering, Chungju National University, 50 Daehak-ro, Chungju-si, Chungbuk 380-702, Republic of Korea

ARTICLE INFO

Article history:
Received 28 October 2010
Received in revised form 17 March 2011
Accepted 13 May 2011
Available online 17 October 2011

Keywords:
Global optimization
Branch-and-bound
B-spline hypervolume
LHS (Latin-hypercube sampling)
Unconstrained minimization
B-spline approximation

ABSTRACT

This paper presents a B-spline-based branch-and-bound algorithm for unconstrained global optimization. The key components of the branch-and-bound, a well-known algorithm paradigm for global optimization, are a subdivision scheme and a bound calculation scheme. For these schemes, we first introduce a B-spline hypervolume to approximate an objective function defined in a design space, where the approximation is based on Latin-hypercube sampling points. We then describe a proposed algorithm for finding global solutions approximately within a prescribed tolerance. The algorithm includes two procedures that are performed iteratively until all stopping conditions are satisfied. One involves subdivision into mutually disjoint subspaces and computation of their bound information, both of which are accomplished by using B-spline hypervolumes. The other updates a search tree that represents a hierarchical structure of subdivided subspaces during the solution process. Finally, we examine the computational performance of the proposed algorithm on various test problems that cover most of the difficulties encountered in global optimization. The results show that the proposed algorithm is complete without using heuristics and has good potential for application in large-scale NP-hard optimization.

1. Introduction

Many research groups, especially those in engineering and science, have made significant efforts to study theoretical and algorithmic aspects of local optimization techniques in the last six decades. In contrast, much less attention has been paid to global optimization algorithms. However, in recent years, we have begun to recognize the presence of many global optimization problems over a wide range of applications and also to understand that existing local optimization methods cannot consistently provide global solutions to those problems. For this reason, global optimization approaches have been attracting considerable interest from research communities in engineering, applied mathematics, and operations research. In addition, remarkable improvements in computational capabilities expand the range of global optimization problems. These advances enable us to tackle many famous hard optimization problems such as those of scheduling, protein folding, packing, nonlinear least squares, chemical equilibrium, and robot arm design.

Existing global optimization algorithms can be classified into *stochastic approaches*, which find the global minimum only with a high probability, and *deterministic approaches*, which are guaranteed to find a global optimum with a required accuracy. The former methods involve function evaluations at a suitably chosen random sample of points and subsequent manipulation of the sample to find global minima. A number of techniques such as simulated annealing [1] and genetic algorithms [2] employ analogies to physics and biology to find a global optimum.

The most successful class of deterministic approaches is branch-and-bound algorithms [3–6]. The branch-and-bound algorithm begins with a branching procedure that subdivides the feasible domain into two or more subspaces. Then a bounding procedure is applied to the subspaces to compute each one's bound information. The branching procedure is applied recursively to the subspaces, generating so-called nodes, which represent unexplored subspaces in a dynamically generated search tree. The lower bounds obtained by the bounding procedure allow us to eliminate large portions of the feasible domain early in the computation, so only a small part of the feasible domain must be processed. The lower bounds can be computed by using such techniques as the difference of convex functions [7] and interval analysis techniques [8]. These methods are generally not applied to all types of objective functions. Only analytic forms are available, since underestimation functions for the lower bounds are derived from the known analytic functions. In addition to the optimization methods described above, we can find internet sites [9,10] containing many commented links to online information and software packages relevant to global optimization, and a good recent online survey of techniques is available [11–13]. This paper proposes a combination of a B-spline-based volume representation and branch-and-bound algorithm to overcome the limitations described above and to achieve fast computation for the lower bounds using the attractive properties of B-spline objects.

* Tel.: +82 43 841 5122; fax: +82 43 841 5120.
E-mail address: skpark@cjnu.ac.kr

0965-9978/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.advengsoft.2011.09.009

In this work, we first introduce a B-spline representation model to describe an objective function with randomly scattered data extracted in a feasible domain, namely, the B-spline hypervolume model [14]. Next, we present its approximation techniques based on the least-squares and pseudo-inverse methods [15]. We then describe the proposed algorithm for finding approximate global solutions to unconstrained optimization problems, which is based on a branch-and-bound framework. The proposed method includes two procedures. One involves branching into subspaces and computing their bound information, both of which are accomplished using B-spline hypervolumes. The other updates a search tree that represents a hierarchical structure of the subdivided subspaces during the solution process; e.g., it prunes subspaces from the search tree. Numerical results on a variety of test examples are presented to illustrate the computational performance of the proposed algorithm, and finally, conclusions are presented regarding its advantages and limitations.

2. B-spline hypervolumes for approximating objective functions

2.1. B-spline hypervolume model

A B-spline hypervolume [14] is a generalized volume model function that represents multivariate scalar fields distributed over a bounded region of multidimensional Euclidean space. It is an extension of B-spline curves or surfaces [16] to multivariate volume objects defined in multidimensional space. Since an objective function is generally a real-valued function of an arbitrary number of design variables, a special version of a B-spline hypervolume can be constructed to represent an objective function defined in D-dimensional design space. Thus, we propose Eq. (1) as a representation model approximating the objective function for global optimization

$$f(x_1,\ldots,x_D)=\sum_{i_1=0}^{n_1-1}\cdots\sum_{i_D=0}^{n_D-1}f_{i_1\ldots i_D}N_{i_1}^{k_1}(x_1)\ldots N_{i_D}^{k_D}(x_D) \quad (1)$$

where x_j is the jth design variable, $f_{i_1\ldots i_D}$ denotes the control value of the objective function, n_j and k_j are the number of control points and the order of the B-spline basis function along the x_j direction, respectively, and $N_{i_j}^{k_j}(x_j)$ denotes the normalized B-spline basis function of order k_j defined on the knot vector in the x_j direction. Knot vectors defined in the D-dimensional design space of Eq. (1) are described as follows:

Knot vector in x_1 direction, $\mathbf{T}_1=\{t_{i_1}^{(1)}\}_{i_1=0}^{i_1=n_1+k_1-1}$

....

Knot vector in x_D direction, $\mathbf{T}_D=\{t_{i_D}^{(D)}\}_{i_D=0}^{i_D=n_D+k_D-1}$

(2)

2.2. Approximation with a B-spline hypervolume

Here we introduce an approximation algorithm that constructs the B-spline hypervolume shown in Eq. (1) with an irregularly scattered data set, i.e., D-dimensional sample points $\mathbf{P}_i=(p_i^1,\ldots,p_i^D)$, and its corresponding function values $Q_i=f(\mathbf{P}_i)$, where $i=0,\ldots,N_p-1$. For simplicity, Eq. (1) is rewritten as

$$f(\mathbf{x})=\sum_{J=0}^{N_c-1}f_J\phi_J(\mathbf{x}) \quad (3)$$

where $\mathbf{x}=(x_1,\ldots,x_D)$, $N_c=n_1\times\cdots\times n_D$, $f_J=f_{i_1\ldots i_D}$, $\phi_J(\mathbf{x})=N_{i_1}^{k_1}(x_1)\times\cdots\times N_{i_D}^{k_D}(x_D)$, and the index J is calculated [14] as follows:

$$J=i_1+(n_1)\times i_2+(n_1\times n_2)\times i_3+\cdots+(n_1\times\cdots\times n_{D-1})\times i_D \quad (4)$$

Approximation algorithms can be classified into two categories: over-determined and under-determined problems. In the former, the number of data points given is not smaller than the number of unknown control values $f_J=f_{i_1},\ldots,i_D$ (i.e., $N_p\geqslant N_c$), and vice versa in the latter case. Both problems can be formulated in the matrix form shown in Eq. (5). When the sample points $\mathbf{P}_I$ and their corresponding function values Q_I are substituted into Eq. (3), we obtain Eq. (5) after some manipulation

$$\{Q_I\}=[\phi_{IJ}=\phi_J(\mathbf{P}_I)]\{f_J\} \quad (5)$$

where $I=0,\ldots,N_p-1$, $J=0,\ldots,N_c-1$, and $[\phi_{IJ}]$ is an $N_p\times N_c$ matrix. When $N_p\geqslant N_c$ (the over-determined problem), we apply a least-squares technique to obtain Eq. (6) for the unknown $\{f_J\}$

$$\{f_J\}=([\phi_{IJ}]^T[\phi_{IJ}])^{-1}[\phi_{IJ}]^T\{Q_I\} \quad (6)$$

When $N_p<N_c$ (the under-determined problem), Eq. (7) can be obtained by applying a pseudo-inverse technique to determine the unknown $\{f_J\}$

$$\{f_J\}=[\phi_{IJ}]^T([\phi_{IJ}][\phi_{IJ}]^T)^{-1}\{Q_I\} \quad (7)$$

The matrices $[\phi_{IJ}]^T[\phi_{IJ}]$ in Eq. (6) and $[\phi_{IJ}][\phi_{IJ}]^T$in Eq. (7) are both symmetric and positive semi-definite, so we can obtain a unique solution $\{f_J\}$. Note that this solution is valid when $[\phi_{IJ}]$ is full rank. If $[\phi_{IJ}]$ is not full rank, then the singular value decomposition technique is used to obtain a solution. Finally, we determine the knot value $t_j^{(e)}$ in the e-directional knot vector, $\mathbf{T}_e=\{t_j^{(e)}\}_{j=0}^{j=n_e+k_e-1}$ ($e=1,\ldots,D$) by the following algorithm [16,17]:

Algorithm: Knot determination for a B-spline approximation

Given $p_{\min}^e\leqslant p_i^e\leqslant p_{\max}^e$ ($i=0,\ldots,N_p-1$), we compute the three parts of the e-directional knot vector, where $e=1,\ldots,D$.

// Step 1: For the front part of the knot vector
We set $t_0^{(e)}=\cdots=t_{k_e-1}^{(e)}=p_{\min}^{e}$.

// Step 2: For the middle part of the knot vector
For $j=k_e$ **to** n_e-1,
$d=(N_p-1)/(n_e-k_e+1)$
$i=\text{int}\,[(k_e-1+j)\times d]$
$\alpha=(k_e-1+j)\times d-i$
$t_j^{(e)}=(1-\alpha)\cdot p_i^e+\alpha\cdot p_{i+1}^e$
End of for-loop

// Step 3: For the rear part of the knot vector
We set $t_{n_e}^{(e)}=\cdots=t_{n_e+k_e-1}^{(e)}=p_{\max}^{e}$.

3. Global optimization algorithm

The branch-and-bound principle provides a general framework for global optimization in non-convex problems. It is non-heuristic in the sense that it maintains provable upper and lower bounds on the globally optimal objective value; it generally terminates with stop conditions guaranteeing that the optimal point found so far is within a prescribed tolerance. However, algorithms based on the branch-and-bound framework can be slow. In the worst case, they require a significant effort that grows exponentially with the problem size; however, in some fortunate cases, they converge with much less effort.

The idea behind the branch-and-bound method is frequently described as a divide and conquers approach in the computer science literature. The main characteristic of this class of algorithms is that the quality of the solution found by the algorithm improves as the