Pruning-based pareto front generation for mixed-discrete bi-objective optimization

SeungBum Hong · Jaemyung Ahn · Han-Lim Choi

Received: 31 May 2003 / Revised: 9 April 2014 / Accepted: 22 April 2014 / Published online: 5 June 2014
© Springer-Verlag Berlin Heidelberg 2014

Abstract This note proposes an effective pruning-based Pareto front generation method in mixed-discrete bi-objective optimization. The mixed-discrete problem is decomposed into multiple continuous subproblems; two-phase pruning steps identify and prune out non-contributory subproblems to the Pareto front construction. The efficacy of the proposed method is demonstrated on three benchmark examples.

Keywords Pareto front · Mixed-discrete optimization · Bi-objective optimization · Pruning · Heuristics

1 Introduction

Consider a bi-objective optimization (BOO) problem whose design vector (x) has both continuous (y = [y1, · · · , y ny]) and discrete (z = [z1, · · · , zn z]) components:

\[
\begin{align*}
\min_{x} J(x) &= \min_{y,z} J(y,z) = \begin{bmatrix} J_1(y,z) \\ J_2(y,z) \end{bmatrix} \\
\text{subject to} & \\
g(y,z) \leq 0, & h(y,z) = 0, \\
y_i \in [l_i, u_i], & i = 1, \ldots, n_y, \\
z_j \in Z_j = \{ z_j^1, \ldots, z_j^{Z_j} \}, & j = 1, \ldots, n_z,
\end{align*}
\]

where g is the inequality constraint vector, h is the equality constraint vector, li and ui are the lower and upper bounds of the ith continuous design variable (yi), and Zj is the set of values that the jth discrete design variable (zj) can take. Let X∗ be the set of design vectors that are Pareto optimal solutions of P:

\[
X^* = \{ x^* \in X | \exists x \in X \setminus \{ x^* \} \text{ s.t. } J(x) \leq J(x^*) \}
\] (1)

where X is the set of feasible design vectors. The problem of Pareto front generation is equivalent to determining X∗.

2 Subproblem decomposition and pruning

2.1 Approach

One way to generate the Pareto front of the original BOO is to divide P into subproblems with specific discrete design vectors and construct X∗ by systematically synthesizing the solutions of the subproblems. First, define the set of discrete design vectors, Z = Z1 × · · · × Zn z, and associated index set K = {1, 2, · · · , |Z|}. Let zk, k ∈ K be the kth element of Z; a subproblem of P associated with this discrete realization, denoted as Pk, can be defined as:

\[
\begin{align*}
\min_{y} J(y,z_k) &= \begin{bmatrix} J_1(y,z_k) \\ J_2(y,z_k) \end{bmatrix} \\
\text{subject to} & \\
g(y,z_k) \leq 0, & h(y,z_k) = 0, \\
y_i \in [l_i, u_i], & i = 1, \ldots, n_y.
\end{align*}
\]

The set of Pareto optimal solutions for Pk is defined as:

\[
X^*_k = \{ x^* \in X_k \setminus \{ z_k \} | \exists y \in Y_k \text{ s.t. } J((y,z_k)) \leq J(x^*) \}
\]
where $\mathcal{Y}_k$ is the set of feasible continuous design vectors of the subproblem.

$\mathcal{X}^*_k$ can be obtained relatively easily using normal boundary intersection (NBI) (Das and Dennis 1998; Marler and Arora 2004; Motta et al. 2012) or the weighted sum (WS) method (Hwang and Masud 2001; Marler and Arora 2004; 2010; Kim and de Weck 2005) combined with reliable nonlinear programming (NLP) solvers. One brute-force way of obtaining $\mathcal{X}^*$ is to first compute $\mathcal{X}^*_k$ for all possible discrete realization $\mathbf{z}_k$ and then identify, among those subproblem solutions, design vectors satisfying (1). But this approach can be computationally intractable if the discrete design space is very large, i.e., large $|\mathcal{Z}|$.

Note that the Pareto optimal solutions for some subproblems may have no common elements with $\mathcal{X}^*$, while the others have common elements with $\mathcal{X}^*$ and thus contribute to constructing the Pareto front of $\mathbf{P}$. Define the index set of irrelevant ($K_{\emptyset}$) and relevant ($K_1$) subproblems, respectively:

$$K_{\emptyset} = \{ k \in K | \mathcal{X}^*_k \cap \mathcal{X}^* = \emptyset \}, \quad K_1 = K \setminus K_{\emptyset}$$

Then, the Pareto optimal solution to $\mathbf{P}$ can be obtained by collecting non-dominated solutions out of the relevant Pareto subproblem solutions:

$$\mathcal{X}^+ = \left\{ \mathbf{x}^* \in \mathcal{X}^*_k \big| \exists \mathbf{x} \in \mathcal{X}^*_k \setminus \{ \mathbf{x}^* \} \text{ s.t. } \mathbf{J}(\mathbf{x}) \leq \mathbf{J}(\mathbf{x}^*) \right\}$$

where $\mathcal{X}^*_k = \bigcup_{k \in K_1} \mathcal{X}^*_k$.

Therefore, if $K_1$ (or equivalently $K_{\emptyset}$) can be identified in advance by some efficient procedure, computational complexity of solving $\mathbf{P}$ will be significantly reduced, in particular, when $|K_1| \ll |\mathcal{Z}|$. This work proposes a set of heuristics to approximately identify $K_1$ by solving constant number of nonlinear programs for each $\mathbf{P}_k$.

2.2 Algorithm

This note presents a mechanism to prune a set of subproblems that are expected not to contribute to construction of the Pareto front of $\mathbf{P}$. The procedure consists of two phases: Phase A based on dominance of subproblem utopia points followed by Phase B based on dominance of center points of subproblem Pareto front.

Phase A-1: Computing subproblem anchor/utopia points

The anchor points of $\mathbf{P}_k$ are obtained as

$$J^u_{k,i} = \mathbf{J}(\mathbf{y}^u_{k,i}) = \begin{bmatrix} J^{a,1}_{k,i} & J^{a,2}_{k,i} \end{bmatrix}, \quad i \in \{1,2\}$$

where $\mathbf{y}^u_{k,i} = \arg\min_{\mathbf{y} \in \mathcal{X}_k} J_i(\mathbf{y}, \mathbf{z}_k)$, i.e., solution to the two sole-objective optimization problems. The utopia point of $\mathbf{P}_k$ is then computed as

$$J^u_k = \begin{bmatrix} J^{a,1}_{1,k} & J^{a,2}_{2,k} \end{bmatrix}.$$

This step computes the anchor points and the utopia points for all $\mathbf{P}_k$ (see five sets of anchor/utopia points Fig. 1).

Phase A-2: Generating a master Pareto front

Cross-checking of dominance between the utopia points allows for identification of $\mathcal{K}^u_k$ that can be used to compute an approximate Pareto front:

$$\mathcal{K}^u_k = \{ k | \exists l \in K \setminus \{ k \}, \mathbf{J}^l \leq \mathbf{J}^u_k \}.$$  \hspace{1cm} (2)

Once $\mathcal{K}^u_k$ is determined, a Pareto front with this subproblem set can be obtained, $\mathcal{X}^*_k$, which is termed master front herein. For example, in Fig. 1, utopia points for $k = 1, 5$ are non-dominated; a master Pareto front is generated by obtaining the solutions of subproblems $\mathbf{P}_1$ and $\mathbf{P}_5$ and selecting non-dominated elements.

Phase A-3: Pruning irrelevant subproblems

For the subproblems not considered in construction of the master front, dominance of those utopia points compared to the master front is investigated to obtain

$$\mathcal{K}^u_{\emptyset} = \{ k | \exists \mathbf{x} \in \mathcal{X}^*_k \text{ s.t. } \mathbf{J}^u_k \geq \mathbf{J}(\mathbf{x}) \},$$

where $\mathcal{X}^*_k \triangleq \bigcup_{k \in \mathcal{K}^u_k} \mathcal{X}^*_k$; the subproblems in this set $\mathcal{K}^u_{\emptyset}$ are pruned. (See in Fig. 1 the utopia point for $\mathbf{P}_2$ is dominated by the master front.)

As a result, at the end of Phase A, subproblems in the set

$$\mathcal{K}^u_1 = K \setminus \mathcal{K}^u_{\emptyset}$$  \hspace{1cm} (3)

are left for consideration of Pareto front generation of $\mathbf{P}$, some of whose subproblem Pareto fronts have already been created in Phase A-2.
Proposition 1. The Pareto front constructed with the subproblem set $K^*_1$ is identical to the true Pareto front $X^*$.

Proof It suffices to prove that any $x_l \in X^*$, $l \in K_0^*$ is dominated by some other design vector; thus, $X^* \cap X^* = \emptyset$. For such $x_l$, $J(x_l) \geq J_l^j$ by the definition of utopia point. The fact that $J_l^j$ is pruned implies that $\exists x_k \in X_{K_0^*} \text{ s.t. } J_l^j \geq J(x_k)$. Therefore, $x_l$ is dominated by at least one element in the master front; thus, it cannot be included in $X^*$. □ □

Depending on the problem type, $K^*_1$ may not be substantially smaller than $K_1$; in this case, the following Phase B procedure can improve the computational efficiency.

Phase B-1: Computing approximate subproblem center points For subproblems that have neither been pruned through Phase A nor used to build the master Pareto front, identify one point on the subproblem Pareto front. This step solves one nonlinear program per subproblem; for instance, if the NBI method is adopted for Pareto front calculation, the following NLP is solved:

$$y_k^c = \arg \min_y J_1(y, z_k)$$

subject to

$$J_2(y, z_k) - J_{a,2}^{1,2} - J_1(y, z_k) - J_{a,1}^{1,1} \leq 0,$$

in addition to the constraints for $P_k$. $X_k^c = [y_k^c, z_k]$ and $J_k^c = J(X_k^c)$ are also computed accordingly.

Phase B-2: Pruning dominated center point A Pareto front for a subproblem can be approximated as an arc passing through the two anchor points and the center point; this step assess the dominance of the subproblem Pareto front based on the piecewise linear segment consisting of these three points. This step checks whether or not the center point is dominated by the master front constructed in Phase A-2; if dominated the associated subproblem is pruned from the candidate list of relevant subproblems:

$$K_{\beta}^* = \{k \mid \exists x \in X_{K_1^*} \text{ s.t. } J_k^* \geq J(x)\}.$$

Observe in Fig. 2 that $P_4$ is pruned as $J_4^c$ is dominated by the master front.

Phase B-3: Generating Pareto front with remaining subproblems Then, the remaining index set becomes

$$K_1^* = K_1^* \setminus K_{\beta}^*,$$

and typical Pareto generation techniques such as weighted sum or NBI method can be used to construct the Pareto front of $P$. As a result, the Pareto front is calculated using the subproblem Pareto fronts in $K_1^*$. In Fig. 2, $P_3$ is included to construct the Pareto front. It should be pointed out that since $K_1^*$ is not a superset of $K_1$, optimality of the resulting Pareto front is not guaranteed with Phase B, in particular when the subproblem Pareto fronts exhibit large curvature or concavity, in contrast to preservation of optimality only with Phase A.

Remark 1. Throughout this two-phase process, the total number of NLPs to be solved to construct the Pareto front is

$$N_{AB} = \frac{2|K_1| + \beta|K_1^m|}{\beta|K_1|} + \frac{(|K_1^m| - |K_1^m|)}{\beta|K_1|} + \frac{\beta(|K_1^m| - |K_1^m|)}{\beta|K_1|}$$

where $\beta$ is the number of points in each subproblem Pareto (the corresponding phase numbers are given beneath the underbraces). Note that without pruning $N_O = \beta|K|$ nonlinear programs need to be solved; thus,

$$\frac{N_{AB}}{N_O} = \frac{2}{\beta} \frac{|K_1^m| + \frac{|K_1^m|}{|K_1|} - \frac{|K_1^m|}{|K_1|}}{|K_1|^2} \leq \frac{2}{\beta} + \Omega(1), \quad \text{if } \frac{|K_1^m|}{|K_1|} \sim O(1), \quad \text{if } \frac{|K_1^m|}{|K_1|} \sim O(1), \quad \text{if } \frac{|K_1^m|}{|K_1|} \sim O(1)$$

where $\epsilon \leq O(0.1)$. Each case represents when (top) neither phases prune out substantial number of subproblems, (middle) only phase B prunes out many indices, and (bottom) phase A cuts out many indices. Typically $\beta \sim O(1)$ to ensure sufficient accuracy of the Pareto front; thus, the pruning method can achieve $O\left(\frac{1}{\beta}\right) \sim O(10)$ times efficiency for typical (i.e., middle and bottom) cases, while in the worst (i.e., top) case it may lead to solve additional $O(\beta)$ NLPs. □
3 Numerical examples

Three numerical examples are considered to demonstrate the effectiveness of the proposed algorithm.

Van Veldhuizen’s Test Problem  One of the Van Veldhuizen’s test suite that is known to exhibit non-trivial Pareto optimal set is considered (Huband et al. 2006). The formulation is as the following:

\[
\begin{align*}
\text{min} & \quad J_1(\mathbf{x}) = \sum_{i=1}^{2} -10e^{-0.2\sqrt{x_i^2 + x_i^4}} \\
\text{subject to} & \quad x_1 \in [-5, 5], \quad x_2, x_3 \in [-5, -4, \ldots, 4, 5].
\end{align*}
\]

While \(x_1\) is continuous, \(x_2, x_3\) are discrete variables each of which can take 11 possible discrete values; \(|\mathcal{K}| = 121\). For comparison, exhaustive search is implemented to obtain the true Pareto optimal points using the NBI method. Table 1 indicates that the proposed method identifies the same set of non-dominated discrete realizations as the true one; significant number of subproblems are pruned out by Phase A.

Nine Bar Truss As shown in Fig 3, consider a truss with nine bars (Mela et al. 2007), when the goal is to minimize two conflicting objectives: (a) the material volume of the truss \(J_1\), and (b) nodal displacement of the node \(N\) \(J_2\):

\[
\begin{align*}
\text{min} & \quad J_1(\mathbf{x}) = \frac{L}{2\pi E} \left\{ \sum_{i=1}^{9} a_i x_i \right\} \\
\text{subject to} & \quad x_i \in [l_i, 10], \quad i = 1, 2, 3 \\
& \quad x_i \in \{1, 5, 10, 15\}, \quad i = 4, \ldots, 9
\end{align*}
\]

The design variables, \(\mathbf{x} = [x_1, \ldots, x_9]\), are the cross-sectional areas of the bars, of which \(x_1, x_2,\) and \(x_3\) are continuous and all others are discrete. The lower bounds for \(x_1\) through \(x_3\) are given as \(l_1 = \frac{1}{3}, l_2 = \frac{1}{2}, l_3 = \frac{1}{2}\). \(L\) is the length shown in Fig. 3, and \(E\) is the Young’s modulus of the bars. The coefficients \(a_i\) and \(b_i\) are \(i^{th}\) elements of the sets \(A = \{1, 1, 1, \sqrt{2}, 1, \sqrt{2}, 1, \sqrt{2}, 1\}\) and \(B = \{4, 1, 1, 8\sqrt{2}, 4, 2\sqrt{2}, 4, 2\sqrt{2}, 0\}\), respectively. This example has a total of \(|\mathcal{K}| = 4^6 = 4,096\) subproblems. The same set of non-dominated discrete realizations as the true one is identified; 80% of the subproblems are pruned by Phase A and then another 18% by Phase B (Table 1 and Fig. 5).

Planetary Campers  A planetary camper concept originally proposed by Draper Lab. and MIT (Fong et al. 2007; deWeck and Simchi-Levi 2006) is considered as one of attractive architectural options for long-term human planetary exploration missions. This concept allocates the habitability and the mobility functions at two physically separate units: an all-terrain vehicle (ATV) and a camper. The ATV tows the camper equipped with only minimal functionalities for habitation and science. By adopting the aforementioned concept, it is expected that the resultant design will have the lower complexity and larger flexibility compared to a single monolithic pressurized rover with both functions (Fig. 4).

The design in this example is inspired by the concept of platform that enables the system to be used in three different planetary environments: Earth, the Moon, and Mars. Platform-based design has been investigated to improve cost effectiveness and ease of maintenance for mass-produced

Table 1 Effects of pruning on number of subproblems

| Case                | Total | True | Phase A | Phase B |
|---------------------|-------|------|---------|---------|
| Van Veldhuizen      | 121   | 4    | 5       | 4       |
| Nine Bar Truss      | 4,096 | 75   | 829     | 75      |
| Planetary Camper    | 512   | 8    | 10      | 10      |

Fig. 3 Illustration of the Nine bar truss.

Fig. 4 Planetary camper concept. (Choi et al. 2007)
products and to reduce design overruns for complex systems (Simpson et al. 2007). For overall planetary exploration program, several different types of rovers for manned Mars missions and lunar missions (as lunar exploration and precursors for Mars), and many tests rovers that will be operated on the Earth will be required as well. Thus, the tradeoff between the optimum performance of each plan-

te (Messac et al. 2002), over the three variants. Two types of constraints, related to geometry (subscript g) and suspension (subscript s) of the camper are considered: the minimum required values for the geometry variables of the habitat shells and the wheels ($g_i^1$ through $g_i^8$), the dimensional coupling between the shell and the wheels ($g_{i,6}^2$ and $g_{i,7}^6$), and fast settling and small peak responses of the suspension system ($g_{i,8}^9$ through $g_{i,11}^9$) are considered.

There are two types of discrete variables: number of wheels, $N_i^j \in \{4, 6, 8, 10\}$, and power source option, $p^j \in \{\text{Solar panel, Fuel cell}\}$. This example has a total of $|K| = 4^3 \times 2^3 = 512$ subproblems, while only eight discrete realizations contribute to the global Pareto front. Observe that with significant number of subproblems are pruned out by Phase A pruning step (Table 1 and Fig. 5).

4 Conclusions

A pruning scheme has been presented to reduce the num-
ber of discrete realizations to be considered for Pareto front generation of mixed-discrete bi-objective optimization. Significant improvement in computational efficiency by the scheme has been verified on numerical examples. Future work includes extension to generic multi-objective cases.

Acknowledgments This work was funded in part by research grant from the National Research Foundation of Korea (2011-0013956) and in part by the KI Project via KAIST Institute for Design of Complex Systems.
References

Alves MJ, Climaco J (2000) An interactive reference point approach for multiobjective mixed-integer programming using branch-and-bound. European Journal of Operational Research 124:478–494

Choi HI, Hong S, Jun M, de Weck OL, Hoffman JA (2007) Platform-based optimal design of planetary campers. In: 7th World Congress of Structural and Multidisciplinary Optimization, Seoul

Das I, Dennis JE (1998) Normal-boundary intersection: A new method for generating the Pareto surface in nonlinear multicriteria optimization problems. SIAM J Optim 8(3):631–657

Deb K, Samir A, Pratap A, Meyarivan T (2002) A fast and elitist multiobjective genetic algorithm: NPGA-II. IEEE Trans Evol Comput 6(2):182–197

Fong A, Hofstetter WK, Hong S, Judnick D, McClosky S, Mellein J, de Weck OL, Hoffman JA, Crawley EF (2007) Design of a platform-based surface mobility system for human space exploration. In:AIAA SPACE 2007 Conference & Exposition, AIAA, Long Beach, CA

Huband S, Hingston P, Barone L, While L (2006) A review of multi-objective test problems and a scalable test problem toolkit. IEEE Trans Evol Comput 10(5):477–506

Hwang CL, Masud AS (2001) Multiple objective decision making, methods, and applications: A state-of-the-art survey. Springer

Kim I, de Weck O (2005) Adaptive weighted-sum method for bi-objective optimization: Pareto front generation. Struct Multidiscip Optim 29(2):149–158

Marler R, Arora JS (2010) The weighted sum method for multi-objective optimization: New insights. Struct Multidiscip Optim 41(6):853–862

Marler RR, Arora JS (2004) Survey of multi-objective optimization methods for engineering. Struct Multidiscip Optim 26:369–395

Mela K, Koski J, Silvennoinen R (2007) Algorithm for generating the Pareto optimal set of multiobjective nonlinear mixed-integer optimization problems. In:AIAA Structure, Struct Dynamics Mater Conf:1–17

Messac A, Martinez MP, Simpson TW (2002) Introduction of a product family penalty function using physical programming. J Mech Des 124:164

Meza JLC, Yildirim MB, Masud ASM (2009) A multiobjective evolutionary programming algorithm and its applications to power generation expansion planning. Eur J Oper Res 195(2):682–699

Motta RS, Afonso SM, Lyra PR (2012) A modified NBI and NC method for the solution of N-multiobjective optimization problems. Struct Multidiscip Optim 46(2):239–259

Sendin J, Exler O, Banga J (2010) Multi-objective mixed integer strategy for the optimization of biological networks. IET Syst Biol 4(3):236–248

Siddiqui A (2006) Reconfigurability in space systems: Architecting framework and case studies. Massachusetts Institute of Technology, PhD thesis

Simpson TW, Siddique Z, Jiao RJ (2007) Product platform and product family design: Methods and applications. Springer

Ulungu E, Teghem J (1994) Multi-objective combinatorial optimization problems: A survey. J Multi-Criteria Decis Anal 3:83–104

Wang L, Singh C (2007) Compromise between cost and reliability in optimum design of an autonomous hybrid power system using mixed-integer PSO algorithm. Capri, Italy, pp 682–689

de Weck O, Simchi-Levi D (2006) Haughton-Mars project expedition 2005 final report. Tech. Rep. NASA TP-20060214196, NASA