



多目的PSO

と制約条件の感度解析を統合した制約あり多目的最適設計問題の解法

メタデータ	言語: jpn 出版者: 公開日: 2018-01-24 キーワード (Ja): キーワード (En): 作成者: 小木曾, 望, 川治, 翔一郎, 小原, 正宜, 石亀, 篤司, 佐藤, 恵一 メールアドレス: 所属:
URL	http://hdl.handle.net/10466/15705

多目的 PSO と制約条件の感度解析を統合した 制約あり多目的最適設計問題の解法*

小木曾 望^{*1}, 川治 翔一朗^{*2}, 小原 正宜^{*3}, 石亀 篤司^{*3}, 佐藤 恵一^{*4}

Constraint Handling in Multiobjective Particle Swarm Optimization Incorporating Sensitivity Analysis on Constraint Condition

Nozomu KOGISO^{*1}, Shoichiro KAWAJI, Masayoshi OHARA, Atsushi ISHIGAME
and Keiichi SATO

^{*1} Department of Aerospace Engineering, Osaka Prefecture University
1-1 Gakuen-Cho, Naka-ku, Sakai, Osaka, 599-8531 JAPAN

A multi-objective particle swarm optimization (MOPSO) is known as a multiple point search-based meta-heuristic approach to find diverse Pareto solutions efficiently for design problem consisting of continuous design variables, but has difficulty to handle the constraint conditions. Overcoming the disadvantage, this study proposes a hybrid algorithm incorporating MOPSO and sensitivity analysis on constrained conditions. When design candidate with constraint violation appears during searching process, the design candidate is moved to the feasible domain using gradient information of the constraint conditions. Then, the design candidate is moved to the feasible boundary using the bi-section method. The forced transferee is applied to the design candidate in MOPSO algorithms. The proposed approach is different from the other hybrid approaches that the Pareto candidates are improved by local searching algorithms. Rather, the approach recovers the violated design candidate to the Pareto candidate by moving to the feasible boundary. This approach is useful for the design problem such as most structural design problems that the constraint sensitivity is easily obtained and that the Pareto sets will exist on the feasible boundary. Through several numerical examples, the diversity and convergency of the Pareto solutions as the performance of the proposed method are investigated.

Key Words : Multiobjective Optimization, Sensitivity Analysis, Constraint Handling, Multiobjective Particle Swarm Optimization, Hybrid Method

1. は じ め に

複数の評価基準を同時に最適化する多目的最適化法のなかでも、進化的最適化法⁽¹⁾は、複数の良好なパレート解を効率よく探索し、目的関数空間に分散させる多点探索法として、幅広く研究されている。本研究では、鳥の群れを模した多点探索アルゴリズムである粒子群最適化 (Particle Swarm Optimization: PSO)⁽²⁾を多目的最適化問題に拡張した多目的粒子群最適化 (Multiobjective Particle Swarm Optimization: MOPSO)^{(3)~(5)}に着目する。PSO は連続変数からなる設計問題に対して、大域的最適解あるいはそれに相当する準最適解を求めるのに適した手法⁽⁶⁾である。PSO を多目的問題に拡張した MOPSO は、パレート解の多様性や収束性の改善などの研究⁽⁷⁾⁽⁸⁾が盛んに行われている。

その一方で、PSO は遺伝的アルゴリズムと同様に、制約条件を明示的に扱わない手法であるため、機械システ

* 原稿受付 2011 年 10 月 12 日

^{*1} 正員, 大阪府立大学大学院工学研究科航空宇宙工学分野 (〒 599-8531 大阪府堺市中区学園町 1-1)

^{*2} 大阪府立大学大学院工学研究科航空宇宙工学分野 (〒 599-8531 大阪府堺市中区学園町 1-1)

^{*3} 大阪府立大学大学院工学研究科電気情報システム工学分野 (〒 599-8531 大阪府堺市中区学園町 1-1)

^{*4} 正員, 株式会社 IHI 基盤技術研究所 (〒 235-8501 横浜市磯子区新中原町 1)

ムの多目的最適設計問題のような制約を有する問題を効率的に解くための手法が提案されている^{(9)~(11)}。その初期には、いったん元の位置に戻してやりなおす方法⁽¹²⁾や制約を逸脱した場合に速度の慣性項をゼロにする操作⁽¹³⁾が提案された。これらは単純で実装が容易であるが、探索が乱数に依存することになるため、探索効率が悪くなる問題がある。他の手法として、ペナルティ関数が広く用いられている⁽¹⁴⁾。しかしながら、ペナルティパラメータによっては探索効率が著しく悪化するなど、パラメータ選択に対する問題がある⁽¹⁵⁾。そのほか、単一目的問題で提案されている手法もある。探索途中で制約領域を逸脱する場合に実行可能領域境界で反射させて実行可能領域に戻す方法⁽¹⁶⁾、制約逸脱度を目的関数として定式化して二目的最適化問題として定式化する方法^{(17)~(19)}が提案されている。

さて、構造設計などの多くの最適設計問題では目的関数や制約条件が連続変数からなる微分可能な関数で定式化され、感度解析も準解析的に求められることが多い。これらを利用する手法として、MOPSO と局所探索法を統合したハイブリッド型のアルゴリズムが提案されている^{(20)~(22)}。これらの手法では、局所探索法はパレート解候補となりうる優れた解候補をさらに改善するために用いられていて、制約条件の扱いに着目しているわけではない。

一方、多目的 GA では、個体の優劣を順序付けするランキング法^{(23)~(24)}が提案されている。これは、制約条件を満足する解を満足しない解に対して優先し、制約を逸脱した解では逸脱度の小さな個体を優位とする順序付けすることで、制約を満足する解を求めようとするものである。この手法では、制約条件をわずかに逸脱しているけれどもパレート解に近いかもしれない個体を活用することができない。

これら既存の手法では、制約条件をわずかに逸脱しているけれどもパレート解に近いかもしれない個体が有効に利用されていない。構造設計問題などの工学問題では、パレート解集合が実行可能領域の境界上に存在することが多いため、そのような個体を有効に利用することで、探索効率が向上する可能性が高い。

そこで、本研究では、制約条件を逸脱した個体を有効に利用するために、制約情報の感度解析と MOPSO を統合したハイブリッド型の MOPSO を提案する。この手法では、MOPSO の繰り返し過程で制約条件を逸脱した個体に対しては、制約条件の感度を用いて実行可能領域に移動させる。この際、繰り返し計算を必要とするが、この移動により、従来の手法では有効に利用されていなかった個体を優れたパレート解候補として利用できるため、広範囲のパレート解を効率的に探索することが期待できる。

以下、2 章では多目的先の設計問題の定式化と MOPSO について簡単に説明する。続いて、3 章で本研究で提案する制約条件の扱い方および全体のアルゴリズムを示す。そして、4 章では、本提案手法の妥当性を検証するため、いくつかのベンチマーク問題に適用した計算例を示し、5 章で結論を述べる。

2. 多目的最適設計問題と MOPSO

2.1 多目的最適設計問題とパレート解

k 個の目的関数、 m 個の制約条件を有する多目的最適化問題は次式で定式化される。

$$\begin{aligned} &\text{Minimize: } f_l(\mathbf{x}), \quad (l = 1, 2, \dots, k) \\ &\text{subject to: } g_j(\mathbf{x}) \geq 0, \quad (j = 1, 2, \dots, m) \\ &\quad \quad \quad x_i^L \leq x_i \leq x_i^U, \quad (i = 1, 2, \dots, n) \end{aligned} \tag{1}$$

$f_l(\mathbf{x})$ は l 番目の目的関数、 $g_j(\mathbf{x})$ は j 番目の制約関数、 $\mathbf{x}$ は設計変数ベクトル、 n は設計変数の数である。また、 x_i^L と x_i^U は i 番目の設計変数に対する側面制約である。

多目的最適化問題では、複数の目的関数を同時に最小化する解は一般的には存在せず、目的関数間にトレードオフ関係が存在する。任意の 2 つの実行可能解 $\mathbf{x}_i, \mathbf{x}_j (i \neq j)$ が $f_l(\mathbf{x}_i) \leq f_l(\mathbf{x}_j), (\forall l = 1, 2, \dots, k)$ のとき、 $\mathbf{x}_i$ は $\mathbf{x}_j$ に優越するという。そして、 $\mathbf{x}_i$ に優越する $\mathbf{x}$ が存在しない場合、 $\mathbf{x}_i$ をパレート解とよぶ。また、 $f_l(\mathbf{x}_i) < f_l(\mathbf{x}_j), (\forall l = 1, 2, \dots, k)$ のとき、 $\mathbf{x}_i$ は $\mathbf{x}_j$ に強い意味で優越するという。そして、 $\mathbf{x}_i$ に強い意味で優越する $\mathbf{x}$ が存在しない場合、 $\mathbf{x}_i$ を弱パレート解とよぶ。2 目的関数におけるパレート解の例を図 1 に示す。

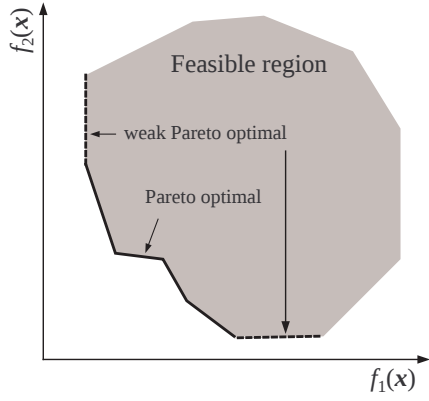


Fig. 1 Pareto optimal solutions

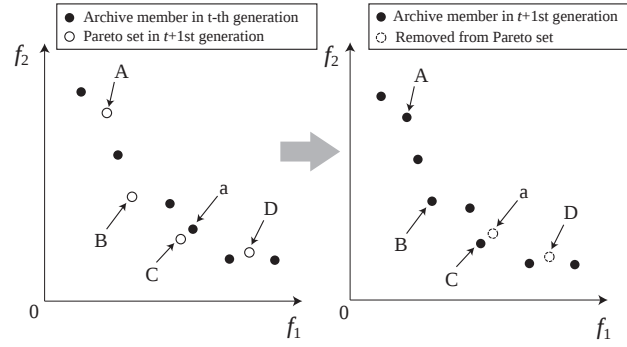


Fig. 2 Pareto solutions stored in archive

2.2 多目的粒子群最適化: MOPSO

MOPSO は、鳥の群れを模したヒューリスティックな多点探索法である PSO を多目的に拡張した多点探索法である。PSO では、 t 回目の繰り返し過程における i 番目の設計変数 $\mathbf{x}_i^t$ を次式によって $\mathbf{x}_i^{t+1}$ に更新する。

$$\mathbf{x}_i^{t+1} = \mathbf{x}_i^t + \mathbf{v}_i^{t+1} \quad (2)$$

$$\mathbf{v}_i^{t+1} = w\mathbf{v}_i^t + C_1r_1(\mathbf{x}_{p_i}^t - \mathbf{x}_i^t) + C_2r_2(\mathbf{x}_{g_i}^t - \mathbf{x}_i^t) \quad (3)$$

ここで、 $\mathbf{v}_i^{t+1}$ は設計変数を更新する速度、 $\mathbf{x}_{p_i}$ は過去の探索における探索点 i の最良値である自己ベスト、 $\mathbf{x}_{g_i}$ は探索点全体における過去の最良値である集団ベストである。 r_1, r_2 は確率論的な探索を行うために設定する $[0, 1]$ の一様乱数である。 w は前回の速度の影響を考慮する慣性項、 C_1, C_2 は $\mathbf{x}_{p_i}, \mathbf{x}_{g_i}$ の影響を考慮するパラメータである。本研究では、安定性を考慮し、一般的に用いられている値 $w = 0.4, C_1 = C_2 = 2$ を用いる⁽⁶⁾。

多目的最適設計に拡張するために、本研究では、非劣解からパレート解候補を選択するエリート保存戦略の一つであるアーカイブ⁽²⁵⁾を用いる。この方法は、図 2 に示すように、現時点で得られたパレート解候補 (A, B, C, D) と既に保存してあるパレート解候補とを比較し、パレート解候補を更新する戦略 (A, B, C の 3 つはパレート解として選ばれ、それまではパレート解であった a は削除される) である。

また、MOPSO は単一目的問題と異なり、式 (3) の $\mathbf{x}_{p_i}, \mathbf{x}_{g_i}$ を一意に決定することができないため、それらを適切に決定する手法が提案されている⁽⁴⁾。本研究では、パレート解の探索効率が高いとされるシグマ法⁽²⁶⁾を用いる。この手法では、まず、目的関数値間の寄与度を示すシグマ値を次式で求める。

$$\sigma = \frac{f_1^2 - f_2^2}{f_1^2 + f_2^2} \quad (\text{if } k = 2) \quad (4)$$

$$\boldsymbol{\sigma} = \frac{1}{\sum_{i=1}^k f_i^2} \begin{pmatrix} f_1^2 - f_2^2 \\ \vdots \\ f_{k-1}^2 - f_k^2 \\ f_k^2 - f_1^2 \end{pmatrix} \quad (\text{if } k \geq 3) \quad (5)$$

$k = 2$ の場合はスカラー、 $k \geq 3$ の場合は k 要素からなるベクトルとして表される。

そして、図 3 に示すように、目的関数空間でシグマ値が近い値をもつパレート解候補を式 (3) の $\mathbf{x}_{g_i}^t$ として選択する。これにより、パレート解の広域性を確保する。

一方、アーカイブに保存されるパレート解の数が増えすぎると図 2 に示したパレート解の比較のための計算量が膨大になってしまう。この問題を解決するため、シグマ値に範囲を設定し、その範囲内で一つのパレート解のみをアーカイブに保存する。これにより、計算回数を抑えつつ、広範囲にわたって均一なパレート解集合を求めることができる。

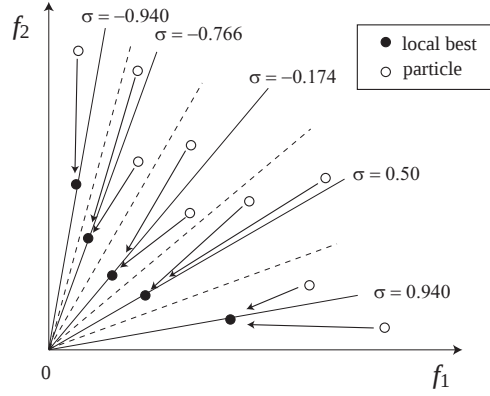


Fig. 3 Local guide selection of sigma method

3. 制約条件の取り扱い

制約条件のある多目的最適設計問題においては、単一目的関数問題と同様に、実行可能領域の境界上に最適解集合が存在することが多い。そのため、実行可能領域の境界上を効率的に探索することが重要となる。特に、機械システムの最適設計問題においては、設計感度を準解析的な方法で求めることができることが多いため、感度情報を有効に利用することで、パレート解を効率的に求められると考えられる。

そこで、本研究では、制約条件を逸脱した個体をその近傍の実行可能領域に囲い込むことで、その個体を有効に活用する手法を提案する。この手法では、制約条件を逸脱した場合、まずは、逸脱した制約条件の感度を用いて、個体の移動方向を決める。そして、その方向に個体を微小量移動させ、逸脱量が改善している場合は、さらに移動量を増大させながら、実行可能領域へ囲い込む。しかし、途中で制約条件関数値が悪化した場合あるいは逸脱した制約条件が変化した場合、その点で制約条件の感度を評価しなおし、改善方向を変更する。すべての制約条件を満足する点が見つければ、その点とその前の制約条件を逸脱している点との2点に対して二分法を適用することで、実行可能領域境界に個体を移動させる。

以下に、そのアルゴリズムを示す。

Step1 準備 制約を違反した設計 $\mathbf{x}_i$ の移動量 α の初期値を設定する。本研究では 0.01 としている。

Step2 移動方向決定 違反している制約関数の勾配 $\nabla g_j(\mathbf{x}_i)$ を用いて、単位移動方向ベクトル $\mathbf{h}$ を決定する。

$$\mathbf{h} = \sum_{j \in J} \frac{\nabla g_j(\mathbf{x}_i)}{|\nabla g_j(\mathbf{x}_i)|} \quad (6)$$

ここで、 J は満足していない制約条件を表す ($J = \{j | g_j(\mathbf{x}_i) < 0, (j = 1, \dots, m)\}$)。

Step3 設計更新 $\mathbf{h}$ 方向に α だけ移動させる。

$$\mathbf{x}_i = \mathbf{x}_i + \alpha \mathbf{h} \quad (7)$$

Step 4 判定 制約条件を評価し、以下のように判定する。

すべての制約条件を満足している場合 Step5 に進む。

制約値が改善している場合 移動量 α を更新し、Step3 に戻る。本研究では、黄金分割比を用いて、 $\alpha = 1.618\alpha$ と探索幅を拡大する。

違反している制約の値が悪化している場合 Step2 に戻り、移動方向 $\mathbf{h}$ を変更する。なお、移動量 α を初期値に戻す。

満足しない制約条件 J が変化している場合 Step2 に戻り、移動方向 $\mathbf{h}$ を変更する。なお、移動量 α を初期値に戻す。

Step 5 制約境界への移動 制約を満足した点と以前の点から二分法を利用して、次式に示すように、いずれかの制約条件が活性となる点を求め、制約条件境界へ移動させる。

$$\min_j g_j(\mathbf{x}_i) \approx 0 \quad (8)$$

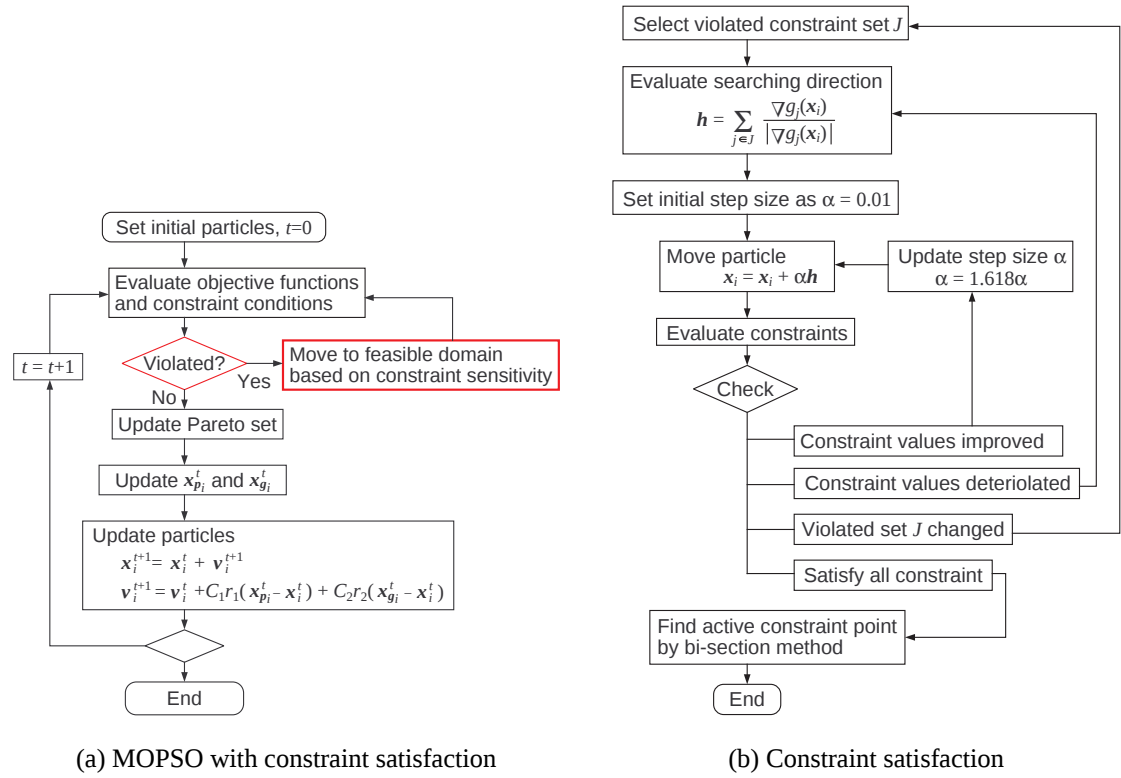


Fig. 4 Computational flow of proposed hybrid MOPSO

この囲い込みおよびこれを組み込んだ MOPSO 全体のアルゴリズムを図 4 に示す。なお、制約条件値の悪影響を避けるために、制約条件を正規化するのが望ましい。

この提案手法では、制約条件を逸脱した個体の実行可能領域境界に移動する際、目的関数や制約条件の計算回数が増加する問題がある。しかし、多くの設計問題においては実行可能領域境界にパレート解が存在することが多いため、パレート解の探索能力が向上することが期待できる。また、初期点が制約条件を逸脱していても実行可能領域境界に移動するため、初期点の設定が容易になる利点がある。

4. 数 値 計 算 例

まず、実行可能領域への囲い込みが機能することを示し、その後、本提案手法をいくつかの例題に適用し、その妥当性を検証する。

4.1 実行可能領域への囲い込み

実行可能領域が凸型および非凸型の組み合わせとして表される次式の問題⁽²⁷⁾を考える。

$$\begin{aligned}
 g_1(\mathbf{x}) &= \frac{x_1^2 x_2}{20} - 1 \geq 0 \\
 g_2(\mathbf{x}) &= \frac{(x_1 + x_2 - 5)^2}{30} + \frac{(x_1 - x_2 - 12)^2}{120} - 1 \geq 0 \\
 g_3(\mathbf{x}) &= \frac{80}{x_1^2 + 8x_2 + 5} - 1 \geq 0 \\
 0 &\leq x_i \leq 10, \quad (i = 1, 2, 3)
 \end{aligned} \tag{9}$$

制約条件を満足しない点からの実行可能領域への囲い込むまでの軌跡を図 5 に、制約条件の評価回数を表 1 に示す。逸脱する制約条件が一つの場合は少ない評価回数で囲い込みに成功している。しかし、(8.5, 1) のように複数の制約条件を逸脱している場合は、 $g_2 = 0$ の境界でジグザグ運動を繰り返して進むために、制約条件の評価回数が多くなってしまう。

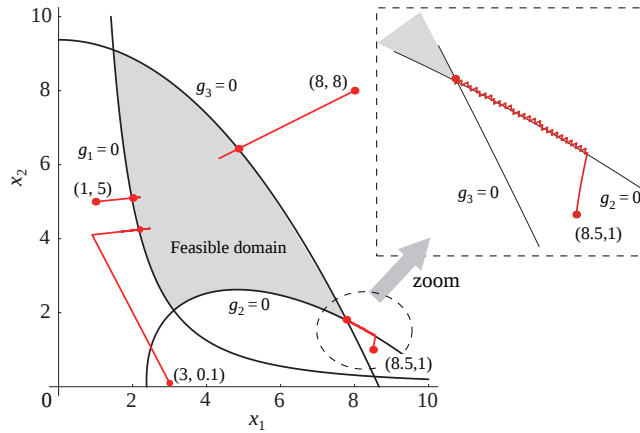


Fig. 5 Feasible domain and example of moving paths to feasible boundary

Table 1 Number of constraint evaluation for enclosure to feasible boundary

Init. pt.	Violated	# Eval.	Conv. pt.
(1,5)	g_1	11	(2.006, 5.101)
(8,8)	g_3	14	(4.342, 6.171)
(3,0.1)	g_1, g_2	24	(2.181, 4.245)
(6,0.1)	g_1, g_2	8	(6.037, 2.377)
(8.5,1)	g_2, g_3	213	(7.777, 1.812)

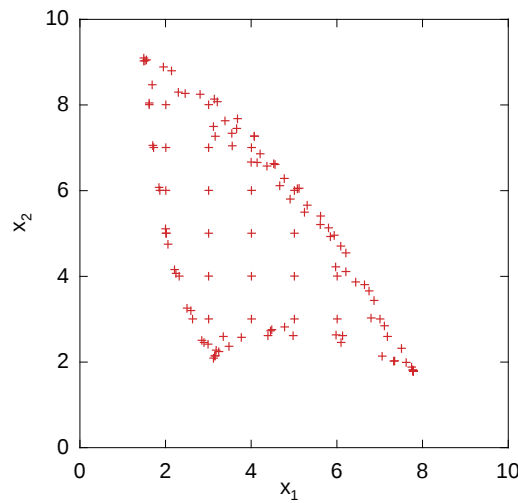


Fig. 6 Point distribution after constraint satisfaction

多くの評価回数を必要とする点がどの程度存在するかを調べるために、個体を定義域内に均等に格子状に配置させて、実行可能領域への移動回数の分布を調べる．図 6 は移動後の点の分布を示す．制約を逸脱した個体を実行可能領域の境界上にほぼ均等に分布するように移動したことが確認できる．このとき、移動に必要な制約条件評価回数は、3 回以下が 18%，10 回以下が 34%，30 回以下が 91%であった．その一方で、1000 回以上必要としたものが約 2%であった．移動によって逸脱する制約が変化し、ジグザグに進むような場合には、多くの評価回数を要するため、このアルゴリズムのさらなる改良が必要である．しかし、ほぼ 9 割の点でその評価回数は十分に満足できるレベルにある．

4.2 制約あり多目的最適化問題への適用

制約あり多目的最適化問題に対する 4 つのベンチマーク問題⁽²⁸⁾⁽²⁹⁾に対して、制約条件を逸脱した個体を実行可能領域に移動させる手法が MOPSO の性能におよぼす効果を調べる．これらの問題において、個体数は 50，繰り返し回数は 500 回と設定している．

4.2.1 TNK 問題

次式で示される 2 変数、2 制約、2 目的関数で表される TNK 問題は、パレート曲線が非凸型で不連続となることで知られている．

$$\text{Minimize : } f_1(\mathbf{x}) = x_1 \quad (10)$$

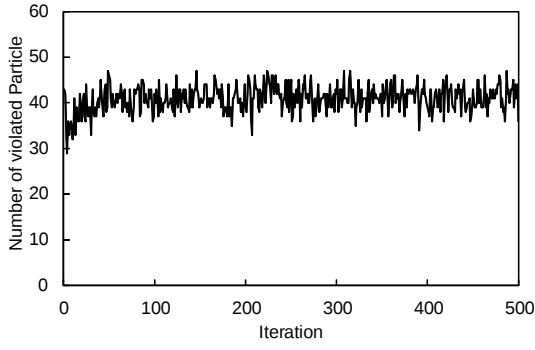


Fig. 7 History of the number of designs with constraint violation in TNK problem

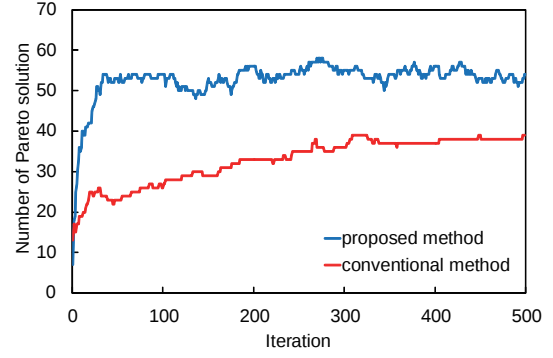


Fig. 8 History of the number of Pareto solutions in TNK problem

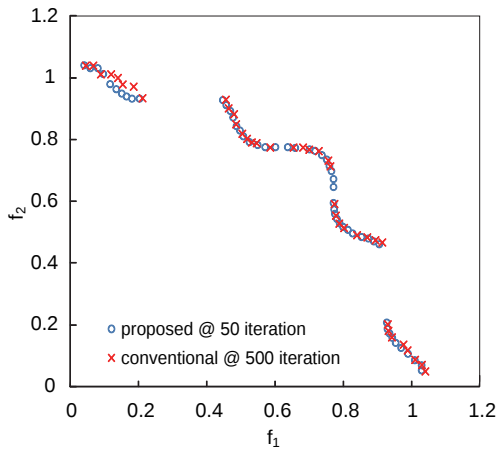


Fig. 9 Comparison of Pareto sets in TNK problem

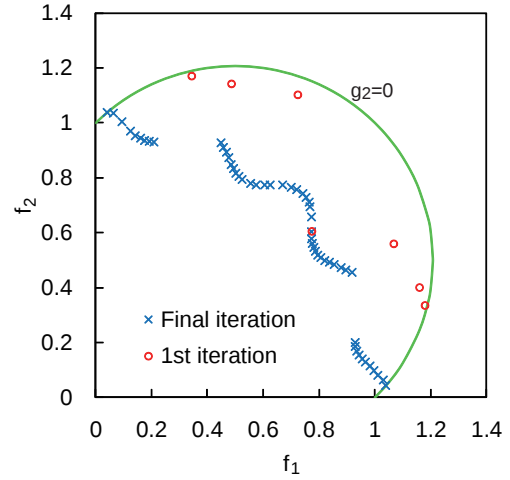


Fig. 10 Pareto sets at the 1st and final generations in TNK problem

$$\begin{aligned}
 f_2(\mathbf{x}) &= x_2 \\
 \text{subject to : } g_1(\mathbf{x}) &= x_1^2 + x_2^2 - 1 - 0.1 \cos \left\{ 16 \tan^{-1} \left(\frac{x_1}{x_2} \right) \right\} \geq 0 \\
 g_2(\mathbf{x}) &= -(x_1 - 0.5)^2 - (x_2 - 0.5)^2 + 0.5 \geq 0 \\
 0 &\leq x_1, x_2 \leq \pi
 \end{aligned}$$

この問題に対して、各世代における制約を逸脱する個数の履歴を図7に示す。世代を通して、約40の個体が制約を逸脱し、感度評価によって実行可能領域に移動されている。これにより、図8に示すように、実行可能領域境界上に存在するパレート解を早い段階から見つけることができ、従来手法と比較して多くのパレート解が見つかったことがわかる。さらに、得られたパレート解の質を図9で比較する。ここに示すように、提案手法の50世代で得られたパレート解と従来手法の500世代で得られたパレート解とを比較し、提案手法の50世代の方が優れたパレート解が得られている。なお、比較対象の従来手法は、制約を逸脱した場合には元の位置に戻して速度をゼロとしてやりなおす方法⁽¹²⁾である。

また、図10では、最初に求められた非劣解と最終的に得られたパレート解を比較する。この問題は $g_1 = 0$ 上にパレート解が存在する。実行可能領域に比べて設計変数の定義域が広いことから、初期に求められた非劣解のうち一つだけが $g_1 = 0$ 上にあるが、ほかの非劣解が $g_2 = 0$ 上または制約が非活性の位置にあるため、最終的なパレート解と一致するものは一つしかないが、早い段階でパレート解が見つかったことがわかる。

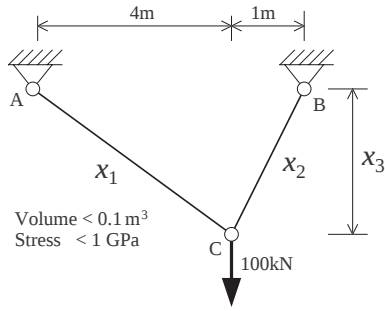


Fig. 11 2-bar truss design problem

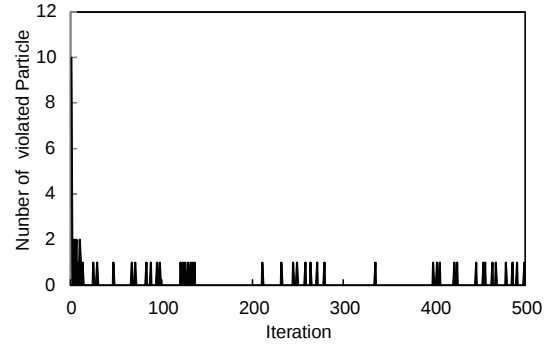


Fig. 12 History of the number of designs with constraint violation in 2-bar truss design problem

4.2.2 2 部材トラス問題

図 11 に示す 2 部材トラス問題では、部材応力および体積制約のもとで、部材体積および部材 AC の応力を最小化する部材断面積を求める問題として、3 変数、3 制約、2 目的関数として定式化される。

$$\text{Minimize : } f_1(\mathbf{x}) = 0.01 \left(x_1 \sqrt{x_3^2 + 16} + x_2 \sqrt{x_3^2 + 1} \right) \quad (11)$$

$$f_2(\mathbf{x}) = 0.2 \frac{\sqrt{x_3^2 + 16}}{x_1 x_3}$$

$$\text{subject to : } g_1(\mathbf{x}) = 1 - f_1(\mathbf{x}) \geq 0$$

$$g_2(\mathbf{x}) = 1 - f_2(\mathbf{x}) \geq 0$$

$$g_3(\mathbf{x}) = 1 - \frac{0.8 \sqrt{x_3^2 + 1}}{x_2 x_3} \geq 0$$

$$0 \leq x_1, x_2 \leq 20, \quad 1 \leq x_3 \leq 3$$

制約を逸脱した個体数の履歴を図 12 に示す。TNK と違ってほとんどの個体は制約を満足していて、図 13 に示すようにパレート解の個数も従来手法と大きな違いはない。この原因として、得られたパレート解のほとんどが、図 14 に示すように、実行可能領域の内側にあることがあげられる。この問題のように、制約条件を満足する解が容易に見つかる問題に対して、提案手法の利点はわかりづらい。

しかし、図 15 に示すように、提案手法の 50 世代でのパレート解と従来手法の 500 世代でのパレート解とを比較すると、提案手法のパレート解の方が優れていることがわかる。この理由として、数こそ少ないが、提案手法により制約が有効なパレート解(図中、塗りつぶしで表している $g_2 \approx 0$ の 1 点と $g_3 \approx 0$ の 3 点)が得られることで、そのほかの優れたパレート解も得られるからだと考えられる。

4.2.3 減速機設計問題

図 16 に示す減速機に対する設計問題は、7 変数、11 制約、2 目的関数からなる問題として次式で定式化される。

$$\begin{aligned} \text{Minimize : } f_1(\mathbf{x}) = & 0.7854x_1x_2^2 \left(\frac{10x_3^2}{3} + 14.933x_3 - 43.0934 \right) - 1.508x_1(x_6^2 + x_7^2) \\ & + 7.447(x_6^3 + x_7^3) + 0.7854(x_4x_6^2 + x_5x_7^2) \end{aligned} \quad (12)$$

$$f_2(\mathbf{x}) = \frac{\sqrt{(745x_4/x_2x_3)^2 + 1.69 \times 10^7}}{0.1x_6^3}$$

$$\text{subject to : } g_1(\mathbf{x}) = x_1x_2^2x_3 - 27 \geq 0$$

$$g_2(\mathbf{x}) = x_1x_2^2x_3^2 - 379.5 \geq 0$$

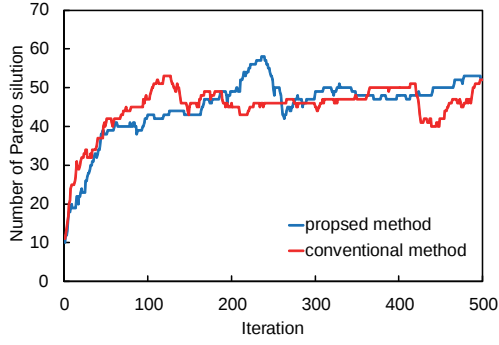


Fig. 13 History of the number of Pareto solutions in 2-bar truss design problem

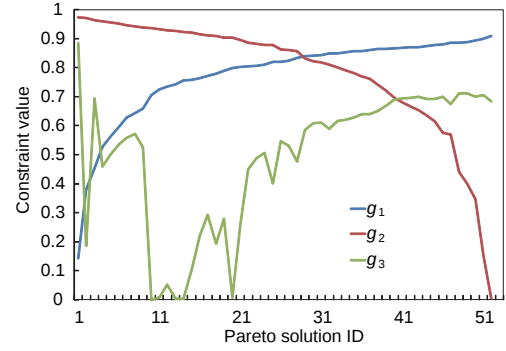


Fig. 14 Constraint values of Pareto solutions in 2-bar truss design problem

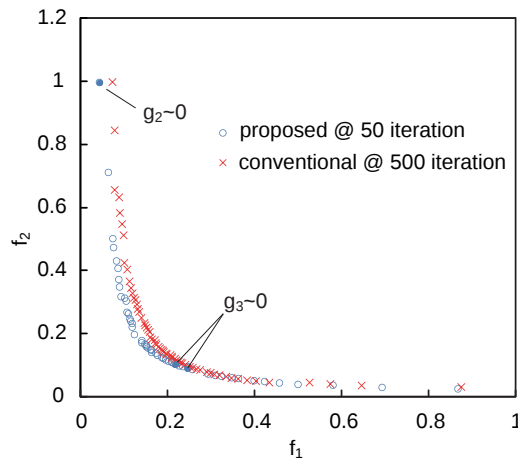


Fig. 15 Comparison of Pareto set in 2-bar truss design problem

$$\begin{aligned}
 g_3(\mathbf{x}) &= x_2 x_3 x_6^4 - 1.93 x_4^3 \geq 0 \\
 g_4(\mathbf{x}) &= x_2 x_3 x_7^4 - 1.93 x_5^3 \geq 0 \\
 g_5(\mathbf{x}) &= 40 - x_2 x_3 \geq 0 \\
 g_6(\mathbf{x}) &= 12 x_2 - x_1 \geq 0 \\
 g_7(\mathbf{x}) &= x_1 - 5 x_2 \geq 0 \\
 g_8(\mathbf{x}) &= x_4 - 1.5 x_6 - 1.9 \geq 0 \\
 g_9(\mathbf{x}) &= x_5 - 1.1 x_7 - 1.9 \geq 0 \\
 g_{10}(\mathbf{x}) &= 1300 - f_2(\mathbf{x}) \geq 0 \\
 g_{11}(\mathbf{x}) &= 110 x_7^3 - \sqrt{(745 x_5 / x_2 x_3)^2 + 1.575 \times 10^8} \geq 0 \\
 2.6 \leq x_1 \leq 3.6, \quad 0.7 \leq x_2 \leq 0.8, \quad 17 \leq x_3 \leq 28, \\
 7.3 \leq x_4, x_5 \leq 8.3, \quad 2.9 \leq x_6 \leq 3.9, \quad 5.0 \leq x_7 \leq 5.5
 \end{aligned}$$

目的関数 f_1 は減速機部品の体積の和, f_2 は軸 1 の応力を表している. 設計変数は, x_1 がギア幅, x_2 がギアモジュール, x_3 が歯数である. x_4, x_5 はそれぞれ, ベアリング 1, ベアリング 2 の隙間, x_6, x_7 はそれぞれ, 軸 1, 軸 2 の直径であり, それぞれ, 上下限界が設定されている. なお, x_3 は本来は整数変数であるが, ここでは実数として扱う. 制約関数は, g_1 がギアの曲げ応力制約, g_2 がギアの接触応力制約, g_3 と g_4 が軸の横変位制約, g_5, g_6, g_7 が寸法制約, g_8, g_9 が軸の設計要求, g_{10}, g_{11} が軸の応力制約を表す.

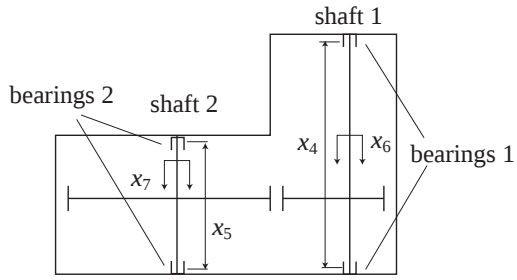


Fig. 16 Speed reducer design problem

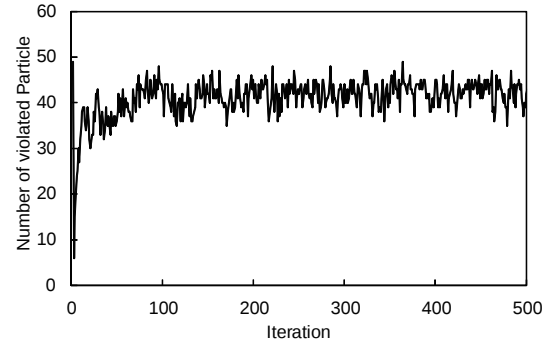


Fig. 17 History of the number of designs with constraint violation in speed reducer design problem

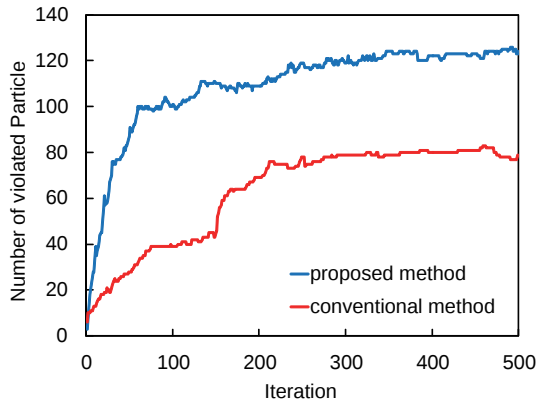


Fig. 18 History of the number of Pareto solutions in speed reducer design problem

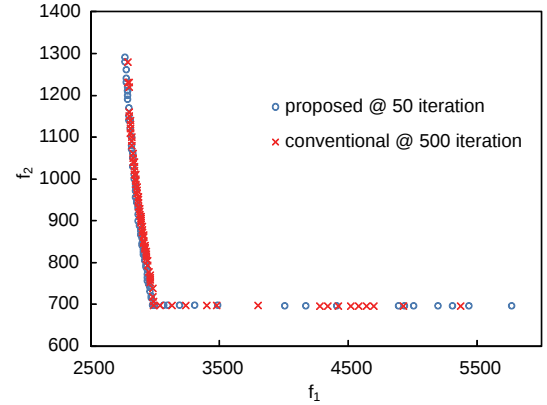


Fig. 19 Comparison of Pareto sets in speed reducer design problem

この問題は、TNKと同様、制約条件を満足する解を求めることが困難であり、図17に示すように各世代で制約を逸脱する個数が40個ほど現れる。提案手法を適用することにより、図18に示すように、従来法と比較して、早い段階で多くのパレート解を見つけることができる。また、図19に示すように、本提案手法の50世代で得られたパレート解集合が、従来法の500世代で得られたパレート解集合よりも広範囲のパレート解を得ていることがわかる。上記の問題と同様に、本提案手法では実行可能領域境界にあるパレート解が容易に得られるためである。

4.2.4 3 目的関数

最後の例として、3 目的関数問題のひとつである DTLZ8 問題 (30 設計変数, 3 制約関数) に適用する。

$$\begin{aligned}
 \text{Minimize : } f_1(\mathbf{x}) &= \frac{1}{10} \sum_{i=1}^{10} x_i \\
 f_2(\mathbf{x}) &= \frac{1}{10} \sum_{i=11}^{20} x_i \\
 f_3(\mathbf{x}) &= \frac{1}{10} \sum_{i=21}^{30} x_i \\
 \text{subject to : } g_1(\mathbf{x}) &= f_3(\mathbf{x}) + 4f_1(\mathbf{x}) - 1 \geq 0 \\
 g_2(\mathbf{x}) &= f_3(\mathbf{x}) + 4f_2(\mathbf{x}) - 1 \geq 0 \\
 g_3(\mathbf{x}) &= 2f_3(\mathbf{x}) + f_1(\mathbf{x}) + f_2(\mathbf{x}) - 1 \geq 0 \\
 0 \leq x_i &\leq 1, \quad (i = 1, \dots, 30)
 \end{aligned} \tag{13}$$

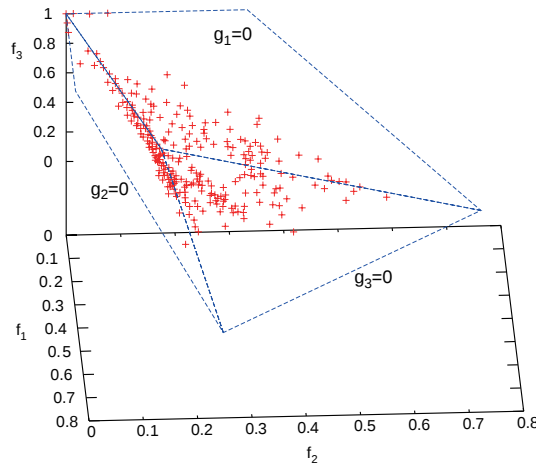


Fig. 20 Nondominate solutions of DTLZ8 problem.

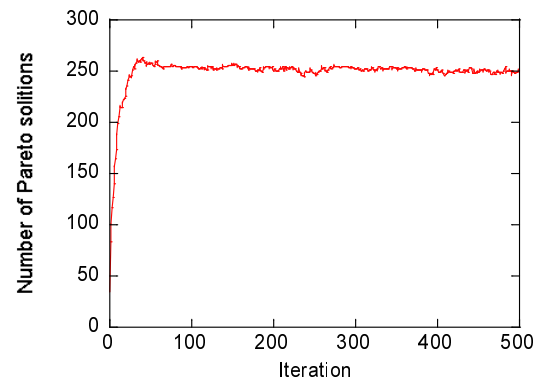


Fig. 21 Number of Pareto solutions in DTLZ8 problem.

Table 2 Performance of proposed MOPSO

Problem	Number of Pareto solutions	Number of function evaluations	Number of constraint sensitivity evaluations	Ratio
TNK	54	368593	54216	16.91
Truss	52	25659	59	1.03
Speed	123	263142	37227	12.01
DTLZ8	251	144222	13891	5.77

得られた 251 個の非劣解を目的関数空間に配置した図を図 20 に示す. この問題のパレート解は, 目的関数空間において, 制約関数が等号成立する $g_1 = 0$ 平面と $g_2 = 0$ 平面の交線と $g_3 = 0$ 平面である⁽²⁹⁾. 本提案手法で得られる非劣解は弱パレート解を避けることができないため, 図 20 で示すように, $g_1 = 0$ 平面および $g_2 = 0$ 平面上の点も得られる.

一方, 他の問題と同様, 図 21 に示すように, 繰り返しの早い段階でほとんどのパレート解が得られていることがわかる.

4.2.5 計算回数の比較

4 種類の例題を通して, 本論文で提案した手法は, 従来手法よりも優れたパレート解を, より早い段階で, より広範囲で見つける性能を有していることを示した.

一方で, 計算回数も重要な指標である. そこで, 各問題におけるパレート解の数, 目的関数および感度の評価回数, 1 世代 1 個体に要した計算回数の平均値を表 2 で比較する. 通常の MOPSO では 1 世代 1 個体に必要な計算回数は 1 回であるが, 提案手法では TNK 問題, 減速機問題においては多くの計算回数を要したことがわかる. これは, 制約を違反した個体を戻す際に複数回の計算を必要とするためである. しかし, 2 部材トラス問題では制約を満足することが容易なため, 計算回数が少なくすむ.

このように, 提案手法では多くの計算回数を要する一方で, 提案手法では従来手法の 10 分の 1 の繰り返しで, より優れたパレート解を求めることができる. これを考慮すると, 実質的な計算回数は従来手法と同程度で, 優れたパレート解を得ることができるため, 従来手法よりも優れた性能を有していると言える.

5. お わ り に

本研究では, 構造設計問題のように, 連続な設計変数で定式化される多数の制約条件を有し, 制約条件の感度が容易に利用できる多目的最適化問題に対して, 効率的に高精度なパレート解が得られる手法として, 制約条件を逸脱した場合に制約条件の感度解析を利用して実行可能領域境界へ移動させる機能を多目的粒子群最適化 (MOPSO) に導入した手法を提案した. 数値解析例を通して, パレート解が実行可能領域境界に存在する問題に対して, 本提案手法が有効に機能することを示した. 以下に, 得られた結果を示す.

(1) 制約条件を逸脱した個体に対して, 制約条件の感度解析および二分法を利用して実行可能領域境界に移動さ

せることにより、探索の早い段階で精度の高いパレート解として利用できる可能性が高くなる。

- (2) 制約を逸脱した場合に、元の位置に戻して速度をゼロとしてやりなおす従来手法と比較して、本提案手法では探索の早い段階で、より高精度なパレート解をより多く求めることができる。そのため、従来の手法と比較して、繰り返し世代数を大幅に削減することが可能である。
- (3) ただし、制約条件を逸脱した個体を実行可能領域に戻すために繰り返し回数が必要となるため、1 個体あたりの計算回数が増加する。特に、実行可能領域境界で複数の制約条件境界が入り組んでいる場合では、実行可能領域境界への移動に非常に多くの計算回数を要することもある。
- (4) これらから、全体の計算負荷は従来の手法と同レベルにあると言える。しかしながら、本提案手法では、実行可能領域境界に存在するパレート解の探索効率が向上するため、従来手法と比較して、得られるパレート解の精度が高い。

文 献

- (1) Deb, K., *Multi-Objective Optimization Using Evolutionary Algorithms*, (2001), Wiley.
- (2) Kennedy, J., and Eberhart, R. C., *Swarm Intelligence* (2001), Morga Kaufmann Publishers.
- (3) Coello Coello, C. A., Pulido, G. T., and Lechuga, M. S., “Handling Multiple Objectives with Particle Swarm Optimization”, *IEEE Transactions on Evolutionary Computation*, Vol. 8, No. 3 (2004), pp. 256-279.
- (4) Reyes-Sierra, M. and Coello Coello, C. A., “Multi-Objective Particle Swarm Optimizers: A Survey of the State-of-the-Art”, *International Journal of Computational Intelligence Research*, Vol. 2, No. 3 (2006), pp. 287-308.
- (5) Durillo, J. J., García-Nieto, J., Nebro, A. J., Coello Coello, C. A., Luna, F., and Alba E., “Multi-Objective Particle Swarm Optimizers: An Experimental Comparison”, *Proceedings of the 5th International Conference on Evolutionary Multi-Criterion Optimization* (2009), pp. 495-509.
- (6) 石亀 篤司, 安田恵一郎, “群れの知能: Particle Swarm Optimization”, 日本知能情報ファジィ学会誌, Vol. 20, No. 6 (2008), pp. 829-839.
- (7) Nebro, A. J., Durillo, J. J., García-Nieto J., Coello Coello, C. A., Luna, F., and Alba, E., “SMPSO: A New PSO Metaheuristic for Multi-objective Optimization”, *Proceedings of IEEE Symposium on Computational Intelligence in MCDM* (2009), pp. 66-73
- (8) Higashitani, M., Ishigame, A., and Yasuda, K., “Particle Swarm Optimization Considering the Concept of Predator-Prey Behavior”, *Proceedings of 2006 IEEE Congress on Evolutionary Computation* (2006), pp. 1541-1544.
- (9) Coello Coello, C. A., “Theoretical and Numerical Constraint-Handling Techniques Used with Evolutionary Algorithms: A Survey of the State of the Art”, *Computer Methods in Applied Mechanics and Engineering*, Vol. 191 (2002), pp. 1245-1287.
- (10) Sienz, J. and Innocente, M. S., “Particle Swarm Optimization: Fundamental Study and its Application to Optimization and to Jetty Scheduling Problems”, Chap. 6, *Trends in Engineering Computational Technology*, Topping, B. H. V. and Papadrakakis, M., Eds. (2008), pp. 103-126, Saxe-Coburg Publications.
- (11) Woldesenbet, Y. G., Yen, G. G., and Tessema, B. G., “Constraint Handling in Multiobjective Evolutionary Optimization”, *IEEE Transactions on Evolutionary Computation*, Vol. 13, No. 3 (2009), pp. 514-525.
- (12) Hu, X., and Eberhart, R., “Solving Constrained Nonlinear Optimization Problems with Particle Swarm Optimization”, *Proceedings on 6th World Multiconference on Systemics, Cybernetics and Informatics* (2002), pp. 203-206.
- (13) Venter, G. and Sobieszcanski-Sobieski, J., “Particle Swarm Optimization”, *AIAA Journal*, Vol. 41, No. 8 (2003), pp. 1583-1589.
- (14) 北山哲士, 荒川雅生, 山崎光悦, “非劣解の多様性を考慮した多目的 Particle Swarm Optimization”, 日本機械学会論文集 C 編, Vol. 74, No. 742 (2008), pp. 1575-1583.
- (15) He, S., Prempan, E. and Wu, Q. H., “An Improved Particle Swarm Optimizer for Mechanical Design Optimization Problems”, *Journal of Engineering Optimization*, Vol. 36, No. 5 (2004), pp. 585-605.

- (16) Bochenek, B., and Foryś, P., "Structural Optimization for Postbuckling Behavior Using Particle Swarms", *Structural and Multidisciplinary Optimization*, Vol. 32 (2006), pp. 521-531.
- (17) Venter, G., and Haftka, R. T., "Constrained Particle Swarm Optimization Using a Bi-Objective Formulation", *Structural and Multidisciplinary Optimization*, Vol. 40 (2010), pp. 65-76.
- (18) Li, L. D., Li, X., and Yu, X., "A Multi-Objective Constraint-Handling Method with PSO Algorithm for Constrained Engineering Optimization Problems", *Proceedings of 2008 IEEE Congress on Evolutionary Computation* (2008), pp. 1528-1534.
- (19) Wei, J., and Wang, Y., "A Novel Multi-objective PSO Algorithm for Constrained Optimization Problems, Simulated Evolution and Learning", *Proceedings of 6th International Conference on Simulated Evolution and Learning* (2006), pp. 174-180.
- (20) Izui, K., Nishiwaki, S., Yoshimura, M., Nakamura, M., and Renaud, J. E., "Enhanced Multiobjective Particle Swarm Optimization in Combination with Adaptive Weighted Gradient-Based Searching", *Engineering Optimization*, Vol. 39 (2007), pp. 981-998.
- (21) Bosman, P. A. N., and de Jong, E. D., "Exploiting Gradient Information in Numerical Multi-objective Evolutionary Optimization", *Proceedings on Genetic and Evolutionary Computation Conference* (2005), p. 755-762.
- (22) He, Q., and Wang, L., "A Hybrid Particle Swarm Optimization with a Feasibility-Based Rule for Constrained Optimization", *Applied Mathematics and Computation*, Vol. 186 (2007), pp. 1407-1422.
- (23) Deb, K., "An Efficient Constraint Handling Method for Genetic Algorithms", *Computer Methods in Applied Mechanics and Engineering*, Vol. 186 (2000), pp. 311-338.
- (24) A. Oyama, K. Shimoyama, and K. Fujii, "New Constraint Handling Method for Multi-Objective and Multi-Constraint Evolutionary Optimization", *Transactions of the Japan Society for Aeronautical and Space Sciences*, Vol. 50, No. 167 (2007), pp. 56-62.
- (25) Knowles, J. D., and Corne, D. W., "Approximating the Nondominated Front using Pareto Archived Evolution Strategy", *Evolutionary Computation*, Vol. 8, No. 2 (2000), 149-172.
- (26) Mostaghim, S., and Teich, J., "Strategies for Finding Good Local Guides in Multi-objective Particle Swarm Optimization", *Proceedings on IEEE Swarm Intelligence Symposium* (2003), pp. 26-33.
- (27) Liang, J., Mourelatos, Z. P., and Tu, J., "A Single-Loop Method for Reliability-Based Design Optimization", *ASME 2004 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, ASME DETC 2004-57255 (2004)
- (28) Gong, W., Cai, Z., and Zhu, L., "An Efficient Multiobjective Differential Evolution Algorithm for Engineering Design", *Structural and Multidisciplinary Optimization*, Vol. 38 (2009), pp. 137-157.
- (29) Deb, K., Thiele, L., Laumanns, M., and Zitzler, E., "Scalable Test Problems for Evolutionary Multiobjective Optimization", *Evolutionary Multiobjective Optimization*, Abaraham, A., Jain, L., and Goldberg, R. Eds. (2005), pp. 105-145, Springer-Verlag.