2004－MPS－49　(2)
2004／5／7

スキーマどん欲法の検討と拡張について

丸山　崇[†]　北　英輔[†]

本研究では，確率的スキーマ貪欲法 (Stochastic Schemata Exploiter, SEE) の特徴を検討し，SSE を拡張した拡張型確率的スキーマ貪欲法 (Extended SSE, ESSE) というアルゴリズムを提案する．SSE は局初解への速い収束特性，制御パラメータの数が少ない単純な制御機構という特徴があるが，ESSE は SSE のスキーマの補正をすることで更なる多様性の維持と到達解，収束率の向上を実現することを目指している．

Investigation and Extension of Stochastic Schemata Exploiter

Takashi Maruyama[†] and Eisuke Kita[†]

This paper investigates Stochastic Schemata Exploiter(SSE) and proposes Extended Stochastic Schemata Exploiter (ESSE). SSE reduces the number of control parameters and can rapidly converge to local optimum. Additionally ESSE has better local optimum. ESSE compernsates schemata of SSE, with the result that ESSE maintains diversity and improves final solution and rate of convergence.

1. はじめに

本研究では確率的スキーマ貪欲法 (Stochastic Schemata Exploiter, SSE)[1] の性能を検討し，SSE を拡張した拡張型確率的スキーマ貪欲法 (Extended SSE, ESSE) というアルゴリズムを提案する．ESSE は同じスキーマが個体集団中に拡散することを防ぎ，多様性を維持することで，SSE より解の精度，収束速度の向上を実現することを目的としている．いくつかの問題において SGA と SSE を比較し，続いてナップサック問題を用いて SSE と ESSE を比較検討する．

2. スキーマ貪欲法[1]

SSE は，集団中のスキーマ H を集団中で H を含む個体の部分集合と対応付け，良いスキーマの選出問題を優れた部分集合の選出問題に置き換えることに着目している．

任意の時点 t において，個体集団 P_t 中の M 個の個体をその評価値の順に並べ，$c_1, c_2, \cdots, c_M$ のようにインデクスをつける．ここで，$x_1, x_2, x_3, \cdots$ は $c_1, c_2, c_3, \cdots$ のストリング (染色体) を示す．一般性を失うことなく最大化問題を仮定し，$f(x_1) > f(x_2) > f(x_3) > \cdots$ とする．c_1 が最良個体である．

スキーマ H の中で一番次数の高いスキーマを抜き出すオペレータを $\Lambda(s(H,t))$ と定義する．c_1 の中で一番次数が高いスキーマは c_1 のストリング x_1 そのものであり，$\Lambda(\{c_1\})$ である．$\frac{f(x_1)+f(x_2)}{2} > f(x_2)$ に注意すると，P_t に含まれるスキーマのうちで 2 番目に位置づけれらるのは，c_1 と c_2 に共通に含まれるスキーマ $\Lambda(\{c_1, c_2\})$ となる．このようにして，スキーマの順位づけ問題は，M 個の個体が作るベキ集合の要素 $\{c_1\}, \{c_1, c_2\}, \cdots$ を，その平均評価値にしたがって並べる問題に置き換えることができる．

SSE のアルゴリズムを以下に示す．

手順 (1)　個体集合の優れたものの中から共通ビットを取り出し，有用なスキーマを抽出する．このために各個体の評価値に基づき個体部分集合を平均評価値の高い順に並べ，上位の部分集合より共通のスキーマを取り出す．

手順 (2)　得られた有用なスキーマに基づき確率的な方法で新たに個体を生成する．このために手順 (1) で得られたスキーマからランダムに個体を生成するとともに，突然変異操作を用いて集団中に多様性を取り込む．

3. 拡張型スキーマ貪欲法のアルゴリズム

拡張型スキーマ貪欲法 (ESSE) は SSE のスキーマ

† 名古屋大学大学院 情報科学研究科 複雑系科学専攻
Graduate School of Information Science ,Nagoya University

の部分集合を補正することで多様性を維持し，解の精度，収束速度の向上を目的としている．

2 つスキーマを比べたとき，(1) 全く同じ関係，(2) 一方がもう一方に含まれる関係，(3) 部分的に一致する関係，(4) 全く異なった関係の 4 つのパターンである．(4) のときはその 2 つのスキーマから特徴のあるスキーマを抽出することは出来ないので，(1)〜(3) の場合について考え，ESSE の処理を定義する．ESSE は以下の 3 つの処理の組合せから構成されるアルゴリズムである．以下でリストとは，部分集合より抽出したスキーマを評価値順にソーティングしたリストである．

3.1 ESSE 処理 1

スキーマ A とスキーマ B が同一のスキーマであるとき，スキーマ A の部分集合とスキーマ B の部分集合から新しい部分集合を作り．一つのスキーマ C として扱う．スキーマ A とスキーマ B は全く同じスキーマなので，新しくできたスキーマ C も 2 つのスキーマと同じスキーマである．

この処理 1 では部分集合を再構築し，平均評価値を計算し直すので，より正確な評価が可能となる．SSE ではリスト中に同じスキーマが存在すると，同じスキーマを集団中に拡散してしまい，多様性が無くなるが，処理 1 を用いることで集団中における同じスキーマの拡散を防ぎ，多様性が維持できると期待できる．

3.2 ESSE 処理 2

スキーマ A とスキーマ B に包含関係があるとき，2 つのスキーマの平均評価値の関係が $\widehat{f}(A) \geq \widehat{f}(B)$ なら，部分集合の再構築をしたスキーマ C(スキーマ A の部分集合とスキーマ B の部分集合の和集合) とスキーマ A の 2 つのスキーマを評価の対象とする．$\widehat{f}(A) < \widehat{f}(B)$ なら，部分集合の再構築をしたスキーマ C とスキーマ B の 2 つのスキーマを評価の対象とする．

処理 2 ではスキーマの包含関係からスキーマを評価し直すことで，解の精度や収束を低下させるようなスキーマを排除すると期待できる．

3.3 ESSE 処理 3

スキーマ A とスキーマ B に共通部分があるとき，スキーマ A とスキーマ B，のうち平均評価値の高いスキーマを評価の対象とする．また，スキーマ A とスキーマ B のスキーマの和集合である共通スキーマ C を生成し，これを評価の対象とする．

処理 3 ではスキーマが似ていればそのスキーマの隣接解を調べるので．近傍探索の効果があると期待できる．

表 1 ESSE の定義
Table 1 Definition of ESSE

Combination	処理 1	処理 2	処理 3
c1	有		
c2	有	有	
c3		有	
c4		有	有
c5			有
c6	有		有
c7	有	有	有
Normal(SSE)			

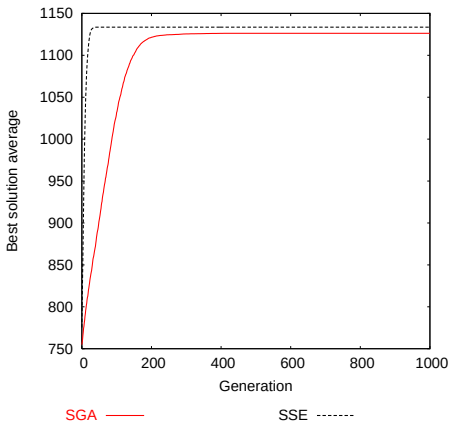

図 1 SGA と SSE の比較 (Deception 問題)
Fig. 1 Comparison of SGA and SSE (Deception problem)

表 2 最終到達解の平均値
Table 2 Average of final solutions

Function	SGA	SSE
Deception	1126.24	1133.58
Knapsack	15407.0	16100.8
Rastrigin	-2.8947	-2.6624
Rosenbrock	-8.4694	-7.7780
Griewank	-0.3197	-0.2331
Ridge	-111.97	-6.24
Schwefel	-64.07	-40.26

3.4 ESSE アルゴリズム

以上に述べた処理 1〜3 の組合せを SSE の手順 (1) 中に導入することで ESSE を定義する．

ESSE の処理 1〜3 の組合せは表 1 に示す 8 通りであるが，処理 1〜3 全てを用いない場合は SSE のアルゴリズムそのものである，よって，それ以外 (必ず 1 つ以上の処理をする) の組合せを ESSE のアルゴリズムとする．

4. SGA と SSE の比較と考察

4.1 解析対象

GA では設計変数間の依存関係の有無や関数の形状

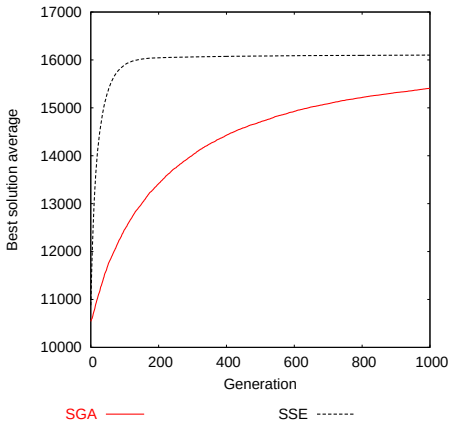

図 2 SGA と SSE の比較 (ナップサック問題)
Fig. 2 Comparison of SGA and SSE (Knapsack problem)

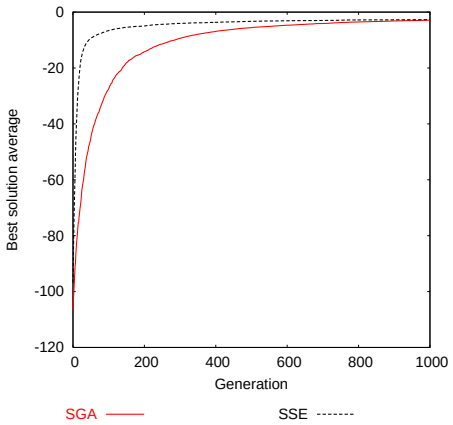

図 3 SGA と SSE の比較 (Rastrigin 関数)
Fig. 3 Comparison of SGA and SSE (Rastrigin function)

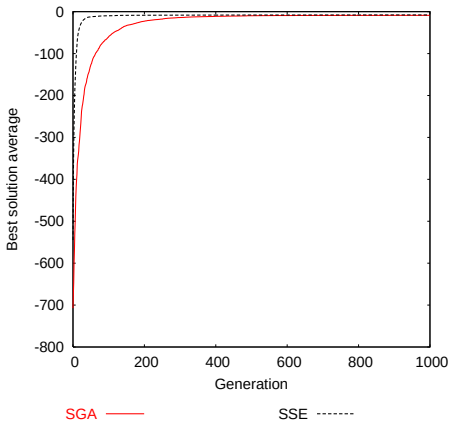

図 4 SGA と SSE の比較 (Rosenbrock 関数)
Fig. 4 Comparison of SGA and SSE (Rosenbrock function)

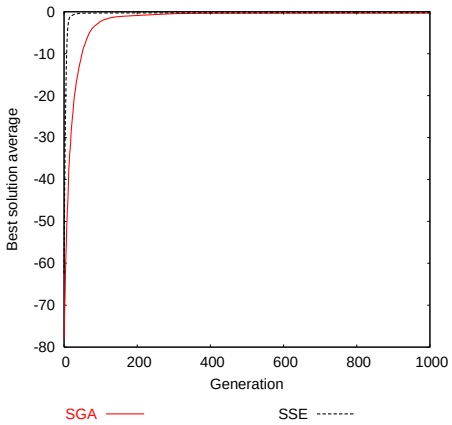

図 5 SGA と SSE の比較 (Griewank 関数)
Fig. 5 Comparison of SGA and SSE (Griewank function)

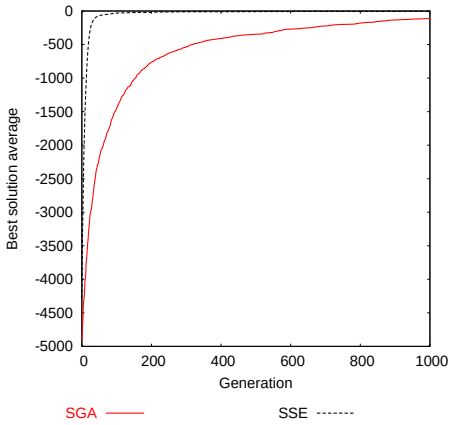

図 6 SGA と SSE の比較 (Ridge 関数)
Fig. 6 Comparison of SGA and SSE (Ridge function)

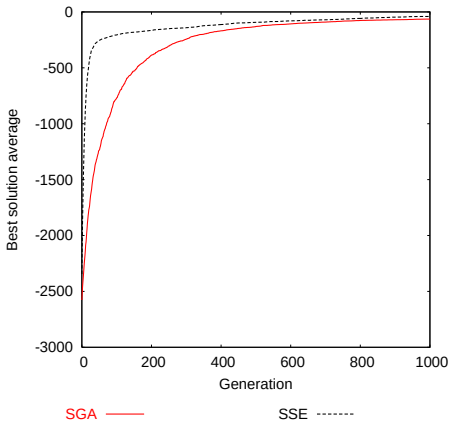

図 7 SGA と SSE の比較 (Schwefel 関数)
Fig. 7 Comparison of SGA and SSE (Schwefel function)

表 3　最終到達解の最良値
Table 3　best values of final solutions

Function	SGA	SSE
Deception	1136.0	1144.0
Knapsack	15592.0	16147.0
Rastrigin	-0.0594	-0.0396
Rosenbrock	-0.1909	-0.7185
Griewank	-0.133	-0.0752
Ridge	-3.0	-2.0
Schwefel	-4.4374	-2.9759

表 4　最終到達解の標準偏差
Table 4　Standerd deviation of final solutions

Function	SGA	SSE
Deception	3.240	5.054
Knapsack	81.18	18.13
Rastrigin	1.834	2.1621
Rosenbrock	9.1184	2.4296
Griewank	0.1060	0.0912
Ridge	147.57	3.77
Schwefel	74.99	58.48

によって問題の性質が異なる．設計変数間に依存関係の有る問題と無い問題の両方について数値実験を行う．本研究では，次の 7 つの問題（関数）を用いて SGA，SSE の性能を比較する．つまり，(1)Deception 問題，(2) ナップサック問題，(3)Rastrigin 関数，(4)Rosenbrock 関数，(5)Griewank 関数，(6)Ridge 関数，(7)Schwefel 関数である．

4.2　解 析 結 果

各関数において 1000 世代実行を計 100 回繰り返したときの突然変異率と最終到達解の平均値の関係を図 1 〜 7 に示す．各グラフにおいて縦軸が到達解の平均値を，横軸が世代数を示す．それぞれの問題での最終到達解の平均値，最良値，標準偏差を表 2,3,4 に示す．

全ての関数において，SGA に比べて SSE は高い収束性を示している．全ての関数において SGA に比べて SSE の標準偏差値は同等もしくは小さいことより，SSE は最終到達解のばらつきがより少ないといえる．ナップサック問題，Rosenbrock 関数，Griewank 関数，Ridge 関数，Rosenbrock 関数，Schwefel 関数の最終到達解の平均値，最終到達解，標準偏差全てにおいて SSE が SGA よりも良い．

5.　SSE と ESSE の比較と考察

5.1　解 析 対 象

次式で定義されるナップサック問題を用いて SSE と ESSE の性能を比較する．

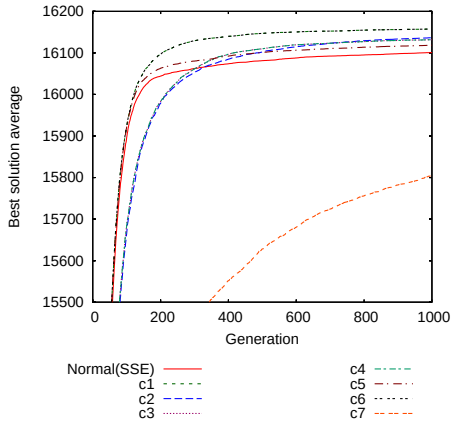

図 8　SSE と ESSE の比較
Fig. 8　Comparison of SSE and ESSE

$$\max_{\{x_i\}} \sum_{i=1}^{n} c_i x_i \qquad x_i \in 0, 1 \quad (i = 1, \cdots, n)$$

$$\text{subject to} \sum_{i=1}^{n} a_i x_i < b$$

ここで，重さ a_i は 0〜100kg の間の値を一様乱数で定め，価格 c_i は 0〜100 円の間の値を一様乱数で定めている．制限重量 b は 10000kg，荷物数 n(次元数) を 400 個とする．

5.2　解 析 結 果

結果を Fig.8 に示す．グラフにおいて縦軸は到達解の平均値を，横軸は世代数を示す．

最終到達解は c1,c6,c2,c3,c4,c5,SSE,c7 の順に良く，収束速度は c1,c6,c5,SSE,c3,c4,c2,c7 の順に良い．

処理 1 を用いる c1,c6 がもっとも良い性能を示したことより，ナップサック問題の解析には処理 1 を適用することで多様性を維持することが解の精度，収束速度の向上を実現することに役立つことが分かった．

c1 と c6, c3 と c4 は最終到達解，収束速度に違いが見られなかった．また，3 つの処理を組み合わせた c7 でも，探索性能は低下している．これらのことより，処理 2 と処理 3 はナップサック問題にはあまり効果的でないといえる．

参　考　文　献

1) 相澤彰子, スキーマ処理に基づく集団型探索アルゴリズムの構成, 電子情報通信学会論文誌, Vol. J78-D-II, pp. 94–104, 1995.
2) D. Whitly, K. Mathias, S. Rana and J. Dzubera. Building Better Test Functions, *Proc. 6th Int. Conf. Genetic Algorithms*. Morgan Kaufmann, 1995.