論文

Quantum-Inspired Evolutionary Algorithm における新たな解探索手法の提案

Proposition of a New Searching Method for Quantum-Inspired Evolutionary Algorithm

今福　啓*1
Kei Imafuku

Email : imafuku@dokkyo.ac.jp

遺伝的アルゴリズムは、メタヒューリスティックな手法として広く利用されている。その解探索に量子コンピュータの計算手順を応用し、性能を向上させた手法に QEA（Quantum-Inspired Evolutionary Algorithm）がある。また、解探索に必要なパラメータ数を減らした手法として QEAPS（Quantum-Inspired Evolutionary Algorithm based on Pair Swap）が提案されている。これらの手法では、解が悪化する方向への探索は確率的にしか実行されず、求められる解が 0 か 1 に限定される。本研究では明示的に解が悪化する方向への探索を取り入れ、解として一般の整数が必要な問題への拡張が可能となる手法を提案する。そして、提案手法を NP（Nondeterministic Polynomial）困難な問題に適用したシミュレーションをおこない、提案手法の解探索の性能について検討する。

GA (Genetic Algorithm) is widely used as a metaheuristic method to solve many problems. To improve the performance of GA, QEA (Quantum-Inspired Evolutionary Algorithm) that uses the procedure of the quantum computer is proposed. In addition, QEAPS (Quantum-Inspired Evolutionary Algorithm based on Pair Swap) has been proposed to reduce the parameters for the solution search. These methods search the solution that makes worse stochastically, and the solution is limited to 0 or 1. In this paper, we propose a method that can explicitly search in the direction where solution becomes worse. The proposed method can be expanded to the problem that requires general integers as a solution. We evaluate the performance of the proposed method by solving the NP-hard problems.

*1：獨協大学　経済学部

1. はじめに

コンピュータが自らさまざまな問題の解を求める際に用いられる手法のひとつに進化計算がある。進化計算は、問題ごとに手法を変えることなく望ましい解を求めることができるメタヒューリスティックスとして知られており、代表的な手法である GA（Genetic Algorithm）をはじめとして、これまで数多くの手法が提案されてきた。

近年注目される手法のひとつとして、量子コンピュータの計算手順を進化計算に応用した QEA（Quantum-Inspired Evolutionary Algorithm）がある[1]。量子コンピュータは、量子力学の原理を利用した並列計算を行うコンピュータである。QEA は、その原理を進化計算手法としてよく利用されている GA に取り入れ、計算効率を向上させた手法となっている。実際、GA と比較してより短い時間での解が発見できることが示されている[2][3]。

QEA では問題の解を表現する個体を複数用意し、それをグループ分けして一定の期間ごとにグループ内、あるいはグループ間で個体のもつ最適解の情報を交換する。QEA は、探索開始の直後はランダムな探索がおこなわれ、時間とともに徐々に最良解の近辺を重点的に探索する SA（Simulated Annealing）に近い手法といえる。

QEA では、パラメータとしてグループ間での交換の期間を設定する必要がある。QEA のもつパラメータを減らし、解探索の効率を向上させた手法として QEAPS（Quantum-Inspired Evolutionary Algorithm based on Pair Swap）が提案されている[3]。QEAPS は個体をグループに分類せず、ランダムにペアリングされた 2 個体間でのみ情報交換をおこなうことで、より良い解の探索が可能であることが示されている。

これらの手法では個体が新たな解を探索する際、個体が生みだす解の候補をその時点での最良解に回転行列を使って一定の角度で近づける。そのため、最良解から離れる方向への探索を明示的に扱わない点に改良の余地が残されていると考えられる[1]。

また、解が 0 か 1 で表現できるビット列に限定されているため、たとえば巡回セールスマン問題のように解が 0, 1 に限定されない一般の整数で表現される問題では、解の表現を問題ごとに工夫することが必要となる。QEA により巡回セールスマン問題の解を求める手法は提案されているが[4][5]、これを解が 0, 1 に限定されない他の一般的な問題にそのまま適用することはできない。

そこで本研究では、QEA をもとにして上記 2 点を改善する手法を提案する。具体的には、解を生み出すパラメータの更新に、回転行列ではなく自動制御の分野で研究されてきた線形時不変の自由系による更新を適用する。これにより、最良解から離れる方向への解の探索を明示的におこなうことが可能となる。そして提案手法および QEAPS を NP 困難な問題として知られているナップサック問題と集合分割問題に適用したシミュレーションをおこない、得られた結果を比較することで提案手法の性能を検証する。

2. QEA

QEA では解の候補となる遺伝子として、解きたい問題に応じて 0 または 1 を任意のビット数（nビットとする）並べて構成する。この遺伝子型は、GA のように 0 か 1 のいずれかに固定されず、量子ビットとよばれる表現を用いて確率的に作成される。0 または 1 の発生確率は、確率振幅とよばれる 2 つのパラメータに依存する。

QEA の各個体は、計算途中の時点で獲得した最良解となる遺伝子型をあわせて記憶する。確率振幅により発生された解の候補が、その時点で持つ最良解よりも望ましい解となった場合、最良解は解の候補で上書きされる。

QEA では、GA とは異なり解の良し悪しによる個体の選択を用いた進化をおこなわない。代わりに各個体がもつ確率振幅を、記憶している最良解を生み出しやすいように更新することで進化させる。

2.1 QEA の計算手順

総数が G_{num} となる個体 i（$i = 1, \cdots, G_{num}$）の持つ最良解候補となる遺伝子型の$k(= 1, 2, \cdots, n)$番目のビットをp_{ik}、最良解のビットをb_{ik}、確率振幅をα_{ik}, β_{ik}とする。p_{ik}, b_{ik}は 0 または 1 のいずれかの値を取る。また、α_{ik}, β_{ik}は次式を満たす。p_{ik}は、以下の手順のとおりα_{ik}^2にもとづいて確率的に 0 または 1 となる。

$$\alpha_{ik}^2 + \beta_{ik}^2 = 1 \tag{1}$$

個体 iがもつ情報$p_i, b_i, \alpha_i, \beta_i$は以下のようになる。

$$p_i = [p_{i1}, p_{i2}, \cdots, p_{in}] \tag{2}$$
$$b_i = [b_{i1}, b_{i2}, \cdots, b_{in}] \tag{3}$$
$$\alpha_i = [\alpha_{i1}, \alpha_{i2}, \cdots, \alpha_{in}] \tag{4}$$
$$\beta_i = [\beta_{i1}, \beta_{i2}, \cdots, \beta_{in}] \tag{5}$$

QEA の計算手順は以下のとおりである。

1. 全ての個体のα_{ik}, β_{ik}を$1/\sqrt{2}$として、p_{ik}の各ビットで 0 または 1 を半々の確率で発生できるように初期化する。
2. 各個体をg個のグループに分ける。
3. 以下の手順 2〜7 を 1 世代として、設定した世代数繰り返す。また一定の世代が経過するごとに手順 8 をおこなう。
4. p_iを作成する。p_{ik}は、0 から 1 の範囲にある一様乱数r_{ik}を発生させた際に$r_{ik} < \alpha_{ik}^2$ならば 0、そうでなければ 1 とする。なお、最初にp_iを作成したときのみ、p_iをb_iとする。
5. p_iが与えられた問題にどの程度適しているのかをあらわす適応度を、問題ごとに用意された関数$f(p_i)$により計算する。
6. $f(p_i) \leq f(b_i)$かつ$p_{ik} \neq b_{ik}$の場合、今後作成するp_{ik}がb_{ik}に近い値となるようα_{ik}, β_{ik}を更新する。更新は、以下の回転行列によりおこなう。こ

[1] 後に述べるとおり、QEA や QEAPS は探索の初期時刻には確率的に最良解から遠ざかる方向への探索が可能であるが、時間経過とともに最良解から離れる方向への探索をおこなわない手法となっている。

こでθ_cは、α_{ik}, β_{ik}の更新速度を決めるパラメータとなる回転角である。値を大きくすれば、一度の更新で大きく変更される。また≔は、右辺で計算した結果を左辺に代入することを意味する。

$$\begin{bmatrix}\alpha_{ik}\\ \beta_{ik}\end{bmatrix} \coloneqq \begin{bmatrix}\cos\theta_c & -\mathrm{sign}\cdot\sin\theta_c\\ \mathrm{sign}\cdot\sin\theta_c & \cos\theta_c\end{bmatrix}\begin{bmatrix}\alpha_{ik}\\ \beta_{ik}\end{bmatrix} \quad (6)$$

$$\mathrm{sign} = \begin{cases} b_{ik} - p_{ik} & (\alpha_{ik}\beta_{ik} \neq 0)\\ +1 \text{ or } -1 & (\alpha_{ik}\beta_{ik} = 0)\end{cases} \quad (7)$$

7. $f(p_i) > f(b_i)$ならば、p_iを新たなb_iとする。
8. グループ間で、個体を局所的移住および大域的移住させる。局所的移住ではT_{local}世代ごとに、1 つのグループの中で最も高い適応度をもつ個体iのb_iを、同じグループに属する他のすべての個体の$b_j(j \neq i)$にコピーする。大域的移住ではT_{global}世代ごとに、全ての個体の中で最も高い適応度となるb_iを、他のすべての個体の$b_j(j \neq i)$にコピーする。

2.2 QEAPS

QEAPS では、QEA における手順 2 のグループ分けをおこなわない。また手順 8 のグループ内、グループ間の移住処理の代わりに 1 世代ごとに 2 つの個体でランダムにペアリングし、互いの持つb_iを交換する。

QEA では局所的移住や大域的移住の際に、特定の個体の最良解がグループ内の個体や集団全体に伝播する。そのため、移住が生じるたびにb_iの多様性が失われ、局所解への収束が生じやすくなると推測される。

QEAPS ではb_iの交換を 2 個体間に限定することで、個体のもつ最良解の多様性を維持できる。この操作によって、高い適応度が得られるα_i, β_iを獲得していない初期世代では個体の持つ情報が集約されず、多様な領域で望ましい解を探索できる。世代が進み適応度の高い解を得られるb_iが生じると、それが集団全体に徐々に伝播し、集団内の多くの個体のα_i, β_iがそのようなb_iを生じやすい値に徐々に近づき、最良解の近辺を集中的に探索することが可能となる。

また、QEAPS はグループ数や移住を行う世代数といったパラメータ設定が不要であり、探索の制度を決定するパラメータ調整がθ_cのみとなることも利点となっている。

3. 提案手法

3.1 提案手法の概要

QEA および QEAPS では、α_{ik}, β_{ik}の更新がθ_cを利用した回転行列により計算される。そのため、p_iがb_iに収束する速度はθ_cの値で決定される。また、b_iに近づく方向のみに進化するため、明示的にb_iから遠ざかるような進化はおこなわない。この速度を一通りにするのではなくさまざまに変更することで、望ましい解を得る速度を改善できると考えられる。

提案手法ではp_iを作成する際、回転行列の代わりに自動制御の分野で用いられる自由系の応答にかんする理論[6]を応用する。自由系とは、変化させたい値を状態$x = [x_1, \cdots, x_n]$（x_i (i = 1, ⋯ , n)は実数）として表現し、それがどのように変化するのかを線形時不変系として計算したものである。状態xが連続的に変化する場合、自由系は次の微分方程式で表現される。

$$\dot{x} = Ax \quad (8)$$

$$A = \begin{bmatrix}a_{11} & \cdots & a_{1n}\\ \vdots & \ddots & \vdots\\ a_{n1} & \cdots & a_{nn}\end{bmatrix} \quad (9)$$

このとき、行列Aの固有値が複素平面の左半平面（固有値の実数が 0 よりも小さい値）に存在するならば、状態が時間とともに 0 に収束することが知られている[2]。本論文では状態の軌道を決める固有値をあらかじめ決定し、それを実現する行列Aを作成することで、状態が目標値に収束する軌道を任意に設定できるようにする。固有値から行列Aを作成する方法は次章のパラメータ設定にて後述する。

提案手法では個体の遺伝子型を確率的に作成する際、α_{ik}, β_{ik}の代わりとして状態$\xi_{ik} = [\xi_{ik1}, \cdots, \xi_{ikl}]$を使用する。$l$は$p_{ik}$と$b_{ik}$で使用する整数値である。解が 0, 1 に限定される場合は、$l = 2$とした 2 次元の状態空間を探索する。これを$l \geq 3$にすることで任意の整数となる解、つまり多次元空間を探索するように拡張できる。なお、本研究では QEAPS と解探索の性能を比較するため$l = 2$となる問題のみを扱う。提案手法を一般の整数値が解となる多次元の問題に適用した場合については今後の課題とする。

個体iのkビット目の状態ξ_{ik}は、次の更新式にもとづいて計算される。

$$\dot{\xi}_{ik} = A_i\xi_{ik} + b_{ik} = \begin{bmatrix}a_{11} & \cdots & a_{1l}\\ \vdots & \ddots & \vdots\\ a_{l1} & \cdots & a_{ll}\end{bmatrix}\begin{bmatrix}\xi_{ik1}\\ \vdots\\ \xi_{ikl}\end{bmatrix} + \begin{bmatrix}b_{ik1}\\ \vdots\\ b_{ikl}\end{bmatrix} \quad (10)$$

行列A_iは個体iを目標状態に収束させる際に使用する遷移行列、b_{ik}は最良解であると同時に状態ξ_{ik}の目標値となる。A_iのすべての固有値が複素平面の原点から半径 1 以内に含まれる場合、状態ξ_{ik}は上記の計算をくり返すことで目標値に収束する。

式(10)は微分方程式となっていることから、本論文では任意に設定した微小時間T_D後のξ_{ik}を直接計算できるよう、次式により離散化したものを使用する。

$$\xi_{ik} \coloneqq P_i\xi_{ik} + b_{ik} \quad (11)$$

$$P_i = e^{A_i t_D} = I + \sum_{j=1}^{\infty}\frac{A_i^j}{j!}T_D^j \quad (12)$$

提案手法では、個体iの更新に使用する行列P_iとしてすべての個体で異なるものを割り当てることで、各世代で望ましい解を得るまでの速度を柔軟に変更できるようにする。

また、QEA および QEAPS ではp_iがb_iに近づく方向のみに更新するが、提案手法ではP_iの固有値によって目標値に漸近するだけでなく、発散させるように設定することも可能である。そのため、各個体は最良解を中心として、各世代で割り当てられたP_iにより目標値に遅く、あるいは速く収束するだけでなく、目標値から発散するような状態の軌道も可能であり、幅広い領

[2] 自由系については、たとえばロボットアームの先端位置を目標位置に安定的に制御するといった自動制御の分野で古くから研究されている。入門的な内容は文献[6]、詳細な理論は文献[7]で述べられている。

域で解を探索することができる。

ここで状態ξ_{ik}は実数ベクトルであるため、そのまま整数値をとるp_{ik}として用いることができない。そこで、$\xi_{ikj}(j=1,\cdots,l)$を次の計算により整数値の選択確率に変換する。

$$\zeta_{ikj} = r_{ik} \cdot \max\left(1, \left|\xi_{ikj}\right|\right) \tag{13}$$

r_{ik}は0から1の範囲の一様乱数である。上式により、ζ_{ikj}は0から1の範囲にある実数値となる。また、ξ_{ikj}が大きいほどζ_{ikj}は大きな値となりやすい。そして、ζ_{ikj}の中で最も大きな値の要素番号から 1 を引いたものをp_{ik}とする[3]。

$$\eta_{ik} = \underset{j}{\operatorname{argmax}}\, \zeta_{ikj} \tag{14}$$

$$p_{ik} = \eta_{ik} - 1 \tag{15}$$

式(15)を$k=1,\cdots,n$において計算することで、i番目の個体における最良解の候補p_iを作成する。

3.2 提案手法の計算手順

提案手法における個体$i(i=1,\cdots,G_{\mathrm{num}})$の更新手順は以下のとおりである。

1. 状態$\xi_{ik}(k=1,\cdots,n)$を$[-0.5, 0.5)$の範囲のランダムな実数に初期化する。
2. 固有値を設定して行列A_iを作成し、指定した微小時間T_Dで離散化して行列P_iを作成する。
3. 以下の手順4〜8を1世代分の計算として、指定した世代数くり返す。
4. 式(13)により状態ξ_{ik}からζ_{ikj}を計算する。そして式(14), (15)からp_{ik}を計算してp_iを作成する。
5. p_iの適応度を問題ごとに用意された関数$f(p_i)$により計算する。
6. $f(p_i) > f(b_i)$ならば、p_iを新たなb_iとする。
7. 式(11)によりξ_{ik}を更新する。QEA や QEAPS とは異なり、$f(p_i) > f(b_i)$の場合でも状態ξ_iを更新する。

4. シミュレーション

提案手法の性能をシミュレーションにより検証する。ここでは、問題の規模が少し大きくなるだけで解の組み合わせが爆発的に増加し、最適解を求めることが困難な NP 困難な問題として知られているナップサック問題および集合分割問題を用いた検証をおこなう。これらの問題の性質は大きく異なり、ナップサック問題ではメタヒューリスティックスが有効な解探索の手法として機能するが、集合分割問題では効率よく機能しないことが指摘されている[8]。

提案手法の性能を、QEA と比較してパラメータ数が少なく調整が容易な QEAPS の結果と比較する。なお、以下であつかう問題は QEA および QEAPS の性能評価をおこなっている文献[11]の一部と同一のものである。この文献[11]では、QEAPS の解探索の性能を QEA、SA、粒子群最適化手法といった他のメタヒューリスティックスと比較している。シミュレーション結果では、多くのパラメータ設定で QEA よりも QEAPS が最適解を得た回数が上回っている。そのため、本研究では提案手法と QEAPS を用いたシミュレーション結果から、両者の解探索の性能を比較して検討する。

4.1 ナップサック問題

ナップサック問題は、複数の品物から容量制限の範囲内で価値の合計が最大となるものを選択し、ナップサックに入れる問題である。実問題として、企業において利益を最大化するプロジェクトを選択する問題や、2 地点間の最短経路を求める問題がある。

本論文では荷物の総数をnとし、荷物$j(=1,2,\cdots,n)$の重量をw_j、価値をc_j、その荷物を入れるかどうかをx_jで表現する。x_jは、荷物をナップサックに入れるとき 1、入れないときは 0 となる。また、ナップサックの容量をbとする。これらを使って問題を定式化すると、以下のようになる。

$$\max \sum_{j=1}^{n} c_j x_j \tag{16}$$

$$s.t. \sum_{j=1}^{n} w_j x_j \le b \tag{17}$$

$$x_j \in \{0,1\} \tag{18}$$

解$x=[x_1,\cdots,x_n]$の適応度$f(x)$は、次の式により計算する。

$$f(x) = \sum_{j=1}^{n} c_j x_j - \gamma \cdot \max\left\{0, \sum_{j=1}^{n} w_j x_j - b\right\} \tag{19}$$

上式の第 2 項は、選択した荷物の容量合計が制約を超えなければ0、超えた場合には超えた分にγをかけた値となる。γを大きな値とすることで、制約を満たさない場合には適応度が大幅に減少する。本研究では$\gamma = 100$とした。

提案手法の評価のため、荷物の総数$n=100$、各荷物の重さと価値を$[1,1000]$の範囲に含まれる一様乱数の整数とした問題を 100 問作成した。それぞれの問題において、ナップサックの容量bは荷物の総重量の半分とした。

この問題の最適解は、動的計画法で効率的に全探索し求めることが可能である[9]。シミュレーションでは、動的計画法で得た最適解と提案手法および QEAPS で得た解がどの程度の回数一致するのか比較することで、提案手法の性能を検討する。

4.2 集合分割問題

集合分割問題は、正の整数からなる集合を要素の合計の差が最小となるように2つに分割する問題である。実問題には、球団のチーム分けや施設配置問題がある。

本論文では正の整数の総数をnとし、その集合を$A=\{a_1,a_2,\cdots,a_n\}$と表現する。これを 2 つの集合A_1, A_2に分ける。また、$a_j(j=1,\cdots,n)$がどちらの集合に含まれるのかをx_jで表現する。x_jはA_1に含まれるとき 1、A_2に含まれるとき-1となる。よって、この問題は次のように定式化できる。

[3] ζ_{ikj}の要素番号は$j=1,2,\cdots$と1からはじまるが、解は0から始まるために 1 を引いている。

$$\min \left| \sum_{j=1}^{n} a_j x_j \right| \tag{20}$$

$$x_j \in \{-1, 1\} \tag{21}$$

解$x = [x_1, \cdots, x_n]$の適応度$f(x)$は、次の式により計算する[4]。

$$f(x) = -\left| \sum_{j=1}^{n} a_j x_j \right| \tag{22}$$

提案手法の評価では、整数の総数$n = 100$、各整数を$[1, 2^{10} - 1]$の範囲の一様乱数とした問題を 100 問作成して使用した。

なお、提案手法およびQEAPS で得られる解は0, 1 に限られるため、式(22)では解0 を-1 に置き換えて計算した。

この問題の最適解を求める手法として、CKK（Korf's complete Karmarkar-Karp）が提案されている[10]。ナップサック問題と同様に、CKK で得た最適解と提案手法および QEAPS で得た解がどの程度一致するのか比較することで、有用性を検討する。

4.3 パラメータ設定

QEAPS では、$\theta_c = 10^3\pi, 10^4\pi, 10^5\pi$としてシミュレーションをおこなった。本論文と同じシミュレーションをおこなう文献[11]では、ナップサック問題においてこの設定で解探索の性能が大きく異なる結果に、集合分割問題においてはほぼ同様の性能となっている。本論文では、提案手法と QEAPS で同一の問題を解いて解探索の性能を比較するため、新たにシミュレーションをおこなった。

提案手法のシミュレーションでは、式(9)の行列$A_i\ (i = 1, \cdots, n)$のそれぞれの固有値（複素数$r_e + i \cdot i_m$）の実部r_eを以下のCase 1, Case 2 の 2 通りの方法でランダムに作成したものを使用した。$rand(x)$は$[0, x)$の範囲にあるランダムな実数を発生する関数である。

Case 1: $r_e = -0.5 \cdot (1 + rand(4))$

Case 2: $r_e = 0.5 - 0.5 \cdot (1 + rand(5))$

Case 1 では、固有値が実部$(-2.0, -0.5]$の一様乱数に設定される。複素平面の左半平面に位置することから、状態は時間とともに目標値に近づく。Case 2 では固有値の実部が$(-2.0, 0.5]$のランダムな値となり、複素平面の左半平面に位置しない場合があることから、状態が目標値から発散する行列が作成されることがある。すべての固有値が発散する値であれば最良解に近づくことはないが、ランダムに発生される実部の値のうち、負となる範囲のほうが大きいことから目標値に近づく行列が多くなる。

式(8)の状態がとる軌道は、固有値により大きく異なる。そのため、提案手法ではA_iの固有値を、Case 1, 2 の実部のみからなる重根と、実部および以下の虚部からなる複素共役のものを半々の確率で作成した。

$$i_m = \begin{cases} 0 & \text{重根} \\ 0.5 \cdot (1 + rand(4)) & \text{複素共役} \end{cases}$$

固有値の実部が負となる重根の場合、状態は目標値を中心とする渦巻き状の円を描き、ゆるやかに収束する。また複素共役となる固有値の場合は、目標値にすばやく漸近するよう収束する。

QEAPS ではα_i, β_iが常に一定の角度θ_cでb_iを発生させやすくなる近づくように更新されるが、提案手法ではA_iの固有値および以下で決定される離散化時間T_Dにより決定される。状態が目標値から遠ざかっていくパラメータを明示的に設定できる点は、提案手法の特徴である。そこで、異なる固有値設定が結果にもたらす影響についてシミュレーションを通じて検証する。

設定した固有値を実現する行列A_iは、次のように作成した（計算の詳細な手順は文献[6]を参照）。

$$A_i = \begin{bmatrix} 0 & 1 \\ a_{21} & a_{22} \end{bmatrix} \tag{23}$$

$$\mathrm{a}_{21} = -r_e^2 - i_m^2,\ \mathrm{a}_{22} = 2 * r_e \tag{24}$$

行列A_iを離散化する時間T_Dは、式(11)における状態ξ_{ik}の変化する速度を決定する。T_Dが小さいならば変化がゆるやかに、大きければ速くなる。そこでT_Dの違いが結果にあたえる影響を分析するため、行列 A_iを作成する際$T_D = 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}$と変更してシミュレーションをおこなった。

QEAPS および提案手法での個体数$G_{num} = 100$、遺伝子長$n = 100$とし、各問題の最適解を求める際、世代数が10^6回に達しても最適解が見つからない場合には解探索を終了した[5]。

本論文ではさまざまなパラメータ設定のもとでシミュレーションを実行することで、提案手法の最適化の性能だけでなく、どのようなパラメータ設定が望ましいのかについて検討する。

4.4 シミュレーション結果

提案手法と QEAPS を用いて、ナップサック問題および集合分割問題の適解を求めるシミュレーションをおこなった。いずれの問題でも異なる 100 問を作成し、各シミュレーションでの最適解の探索に使用した。なお、どちらの手法もメタヒューリスティックスであるため、各問題のシミュレーション時に変更したのは式(19)(22)の適応度のみである。プログラムの作成には Go 言語（go version go1.9.2 darwin/amd64）を使用した。

最初に、ナップサック問題の結果を示す。表 1 は、提案手法において離散化時間T_Dをさまざまに変更した際に得られた結果である。表において「回数」は 100 問中で最適解が得られた回数、「最小」「最大」はそれぞれ最適解が求まるまでに経過した世代数の最小値と最大値、「平均」は最適解が求まるまでの世代数の平均

[4] 作成したプログラムでは、適応度を最大化するように最適化をおこなうため、マイナスの符号をつけて差が最小のときに適応度が最大となるようにした。

[5] QEAPS によるシミュレーションがおこなわれている文献[11]には、シミュレーションの回数について「個体の適応度を計算する回数が$\eta_{\mathrm{FE}} = 1.0 \times 10^7$回に達した時点で探索を終了した」と記述されている。100 個体の適応度を計算しているため、探索する世代数に換算すると10^5世代と推測される。このとき個体のすべてを評価すると、適応度の評価回数は$100 \times 10^5 = 10^7$回となる。本研究では個体の評価を 10 倍の回数実行している。

値である[6]。

表 1 の (a) ～ (d) は、それぞれ $T_D = 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}$ とした場合の結果である。

提案手法では、T_D とが小さくなるにつれて最適解を得られる回数が増えている。結果はいずれのパラメータ設定でも 100 問すべてで最適解を獲得する結果は生じなかった。$T_D = 10^{-5}$ の Case 2 の結果が最もよく、99 問で最適解を得た。

提案手法の Case 1 と Case 2 の結果を比較すると、$T_D = 10^{-2}$ 以外では Case 2 における最適解の獲得回数が Case 1 を上回っている。特に $T_D = 10^{-3}$ の結果をみると、Case 1 では 71 回であるのに対し、Case 2 では 85 回と大幅に改善されている。

最適解を獲得するまでの平均回数は、$T_D = 10^{-4}$ 以外では Case 1 と Case 2 で大きな違いはない。$T_D = 10^{-4}$ のみ、Case 1 の平均が 38721 回であるのに対し、Case 2 は 72009 と大幅に増えている。

表 2 は、QEAPS2 のパラメータを $\theta_c = 10^{-3}\pi, 10^{-4}\pi, 10^{-5}\pi$ とした場合の結果をあらわしている[11]。QEAPS では、いずれの θ_c においても 100 問すべてで最適解が得られた。$\theta_c = 10^{-5}\pi$ の結果は文献[11]では 1 度も最適解を得られないという結果となっているが、本研究ではシミュレーションの最大世代数を文献[11]の 10 倍としたことから 100 問すべてで最適解を得られたと考えられる。

最適解を獲得するまでの世代数の平均は、提案手法、QEAPS ともに T_D, θ_c が小さくなるにつれて平均が増加している。QEAPS で最適解を獲得するまでの平均世代数が最も少ない $\theta_c = 10^{-3}\pi$ では 7716 回であるのに対し、提案手法で最適解を最も多く獲得できた $T_D = 10^{-5}$ とした Case 2 の結果では 322067 回と、非常に多くの世代数を必要としている。

表 1 ナップサック問題における提案手法の結果

(a) $T_D = 10^{-2}$

	Case 1	Case 2
回数	34	33
最小	545	605
最大	1225	3565
平均	753	998

(b) $T_D = 10^{-3}$

	Case 1	Case 2
回数	71	85
最小	3854	4052
最大	22989	7677
平均	5234	5151

(c) $T_D = 10^{-4}$

	Case 1	Case 2
回数	92	94
最小	32166	156
最大	48847	226174
平均	38721	72009

(d) $T_D = 10^{-5}$

	Case 1	Case 2
回数	98	99
最小	274502	256115
最大	395152	415696
平均	320740	322067

表 2 ナップサック問題における QEAPS の結果

	θ_c		
	$10^3\pi$	$10^4\pi$	$10^5\pi$
回数	100	100	100
最小	6005	46242	123973
最大	10288	63413	173847
平均	7716	53998	147047

集合分割問題における提案手法の結果を表 3 に示す。(a)～(d)は、表 1 と同様に $T_D = 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}$ とした結果である。いずれの結果でも 100 問すべてで最適解を得ている。最適解を獲得するまでの世代数の平均回数は、どの結果も Case 1 と Case 2 で大きな違いはみられない。また T_D が小さくなるにつれて、最適解を獲得するまでの平均世代数は大きく増加している。

表 4 は QEAPS で $\theta_c = 10^{-3}\pi, 10^{-4}\pi, 10^{-5}\pi$ とした場合の結果である。どの θ_c においても、100 問すべてで最適解を獲得している。文献[11] ではすべての問題で最適解を得る結果とはなっていないが、本研究では最大世代数を文献[11]の 10 倍としたことから異なる結果になったと考えられる。

最適解を獲得するまでの平均回数は、ナップサック問題と異なり θ_c により大きな違いはない。提案手法の平均回数は、ナップサック問題と同様に T_D が増加すると獲得までの平均世代数も増加している。

表 3 集合分割問題における提案手法の結果

(a) $T_D = 10^{-2}$

	Case 1	Case 2
回数	65	70
最小	305	1256
最大	65100	84201
平均	17298	19090

(b) $T_D = 10^{-3}$

	Case 1	Case 2
回数	100	100
最小	424	78
最大	219204	146628
平均	41254	37687

[6] 最小、最大、平均値は、10^6 世代のうちに最適解が見つからなかった場合の結果を除いて計算した。

(c) $T_D = 10^{-4}$

	Case 1	Case 2
回数	100	100
最小	2017	156
最大	246637	226174
平均	72009	65285

(d) $T_D = 10^{-5}$

	Case 1	Case 2
回数	100	100
最小	1068	780
最大	565503	566231
平均	205679	211939

表 4 集合分割問題における QEAPS の結果

	θ_c		
	$10^{-3}\pi$	$10^{-4}\pi$	$10^{-5}\pi$
回数	100	100	100
最小	785	704	704
最大	340729	406620	282906
平均	52797	61490	60332

4.5 考察

提案手法によるナップサック問題の結果を考察する。表 1(a), (b)をみると、10^6世代まで計算しているにもかかわらず、$T_D = 10^{-2}$では最適解を獲得するまでの世代数の最大値が Case 1, Case 2 のそれぞれで 1225, 3565、$T_D = 10^{-3}$では 22989, 7677 と大幅に下回っている。T_Dが大きい場合、探索の初期に発見した局所解に急速に近づく。そのため、その局所解の近辺に最適解が存在すればただちに発見できるが、そうでなければ最適解を発見できないと考えられる。

探索の範囲は、行列A_iの極配置によっても決定される。しかし$T_D = 10^{-2}$と大きな値が設定されると、状態の変化がその影響を上回る速度となり、探索を困難にすると推測される。

T_Dが小さくなると、探索時点で発見した最良解に漸近する速度も遅くなることから、さまざまない領域での探索が可能となり最適解を発見しやすくなる。ただし、$T_D = 10^{-5}$の結果である表 1(d)からわかるとおり、最適解の探索に必要な世代数が増大する。

T_Dが大きい際には最適解の獲得が速く、小さい際には探索性能が向上する結果となったことから、その両方を取り入れることで生じるシミュレーションから結果を分析する。このシミュレーションでは$T_D = 10^{-5}$~10^{-4}の範囲の一様乱数として異なる値を発生させ、それを式(12)の$P_1, P_2, \cdots$を計算する際に別々に割り当てた。結果は表 5 のとおりである。Case 2 では 100 問すべてで最適解を獲得し、かつ平均値も表 1(d)の$T_D = 10^{-5}$の結果と比較して減少している。このように探索の性能が向上することから、今後T_Dを行列P_iを計算するごとに異なる値にすることの有効性を検討する必要である。

表 5 ナップサック問題において、$T_D = 10^{-5}$~10^{-4}の一様乱数とした結果

	Case 1	Case 2
回数	95	100
最小	52871	52011
最大	82800	95561
平均	66635	66771

集合分割問題は、提案手法、QEAPS のいずれにとっても最良解を得やすい問題だったことが結果からみてとれる。この問題では、表 3(a)のように$T_D = 10^{-2}$と大きい場合にも、少ない世代数で効率的な解探索が可能となる。これは提案手法の利点といえる。

得られた結果から、提案手法では Case 1 よりも Case 2 の設定が効率的な問題解決に有効であると考えられる。つまり行列A_iの固有値の実部を複素平面の左半平面として式(8)が安定となるように設定するのではなく、右半平面にも配置して目標値から発散するパラメータ設定が解探索の性能を向上させるといえる。ただし状態の変化する速度はT_Dだけでなく、行列A_iの固有値によっても決定される。固有値を原点から左に遠ざけるほど収束が速まるため、望ましいパラメータ設定については引き続き検討が必要である。

提案手法と QEAPS の結果を比較する。QEAPS はさまざまな問題においてパラメータ変更の影響を受けず最適解を得られる手法といえる。提案手法では、集合分割問題ではすべての問題で最適解を得られるものの、ナップサック問題では表 1 のとおり$T_D = 10^{-4}$~10^{-5}の範囲の一様乱数とした場合でのみ 100 問すべての最適解を得ている。このように、提案手法はパラメータ設定の難しさが問題点として残る。これは、提案手法において調整が必要となるパラメータ数が多いことが影響していると考えられる。

5. おわりに

本論文では、量子コンピュータの計算手順を進化計算に応用したメタヒューリスティックスな手法である QEA および QEAPS の解探索を改善する手法を提案した。具体的には、線形時不変系の自由応答を応用することで、解探索の際に最良解に近づく方向だけでなく、離れる方向を探索することを明示的に可能とした。提案手法は、QEA や QEAPS のように解の表現を問題ごとに工夫することなく、解が 0, 1 のみに限定される問題ではない一般の整数が必要な問題の解探索に応用できる。

提案手法の性能を検討するため、NP 困難なナップサック問題と集合分割問題に適用したシミュレーションを実行した。得られた結果から、提案手法で自由系が発散する、つまり最良解から離れる方向を明示的に探索することが性能を向上させることを示した。また、ナップサック問題ではパラメータ設定により QEAPS と同じく全ての問題で最適解を得られ、集合分割問題では QEAPS よりも短い世代数で最適解を得られることを示した。

今後の課題として、問題の最適解を効率的に求めるために必要なパラメータ設定をどのようにおこなうのかについての検討があげられる。さらに、解を 0, 1 ではなく一般の整数で表現することが必要な TSP のような問題に適用し、提案手法の有効性を検討したい。

参考文献

(1) Han, K. H and Kim, J. H "Quantum-inspired evolutionary algorithm for a class of combinatorial optimization", *IEEE Trans. Evolutionary Computation*, Vol. 6, No. 6, pp. 580-593 (2002)

(2) Han, K. H and Kim, J. H "On setting the parameters of QEA for practical applications: Some guidelines based on empirical evidence", *Genetic and Evolutionary Computation - GECCO 2003*, pp. 427-428 (2003)

(3) 中山、今別府、小野、飯村 "量子風進化的アルゴリズムにおける対交換戦略の検討", 電子情報通信学会誌, Vol. J89-D, No. 9, pp.2134-2139 (2006)

(4) Moriyama, Y., Iimura, I., Ohno, T., and Nakayama, S. "An experimental study on optimization in permutation spaces by quantum-inspired evolutionary algorithm using quantum bit representation", *J. Signal Processing*, Vol. 19, No. 6, pp. 227-234 (2015)

(5) 森山、飯村、大野、中山 "k-Opt 法的局所改善による量子ビット遺伝子表現法を用いた順列最適化", 人工知能学会論文誌, 31 巻, 6 号, pp. 1-10 (2016)

(6) 鈴木、板宮 「例題で学ぶ現代制御の基礎」, 森北出版株式会社 (2011)

(7) 児玉, 須田 「システム制御のためのマトリクス理論」, コロナ社 (1978)

(8) Ruml, W. "Stochastic approximation algorithms for number partitioning", Technical Report TR-17-93, Harvard University, Cambridge, MA, USA (1993)

(9) 紀平、春日 「アルゴリズムとデータ構造 第 2 版」, Softbank Creative (2011)

(10) Korf, R. E "A complete anytime algorithm for number partitioning", *Artificial Intelligence*, Vol. 106, No. 2, pp. 181-203 (1998)

(11) 今別府, 小野, 森重, 黒瀬, 中山 "Quantum-Inspired Evolutionary Algorithm における移住操作と対交換操作の比較検討", 人工知能学会論文誌, 24 巻, 2 号, pp.250-262 (2009)