情報処理学会論文誌　数理モデル化と応用　Vol.7　No.1　84–93 (Mar. 2014)

量子アルゴリズムで用いられるSpan Programの進化計算による導出

佐多 恵悟[1,a)]　松山 開[1,b)]　坂口 裕一[1]　中山 茂[1]　小野 智司[1]

受付日 2013年9月11日，再受付日 2013年10月21日，
採録日 2014年1月17日

概要：近年，Span Program に基づく論理式評価の量子アルゴリズム（Span-Program-based Quantum Algorithm: SPQA）が注目されている．SPQA に適した量子クエリ計算量が少ない最適な Span Program の導出は，一般的な手法が見つかっておらず，対象となる論理式ごとに専門家が試行錯誤的に導出している．特に，入力ビットが多い論理式では行列の要素数が指数関数的に増加するため，導出が困難である．本研究では，量子クエリ計算量が少ない最適な Span Program の導出を最適化問題として定式化し，進化計算を用いて近似解を導出する手法を提案する．

キーワード：量子アルゴリズム，Span Program，進化計算，論理式評価，量子クエリ計算量

Derivation of Span Program for Span-program-based Quantum Algorithm by Evolutionary Computation

Keigo Sata[1,a)]　Haruki Matsuyama[1,b)]　Hirokazu Sakaguchi[1]
Shigeru Nakayama[1]　Satoshi Ono[1]

Received: September 11, 2013, Revised: October 21, 2013,
Accepted: January 17, 2014

Abstract: In recent years, Span-Program-based Quantum Algorithm (SPQA) for evaluating Boolean formulas has been paid attention. However there has been no general method to derive optimal span program, which make the quantum query complexity of SPQA the least, and only professionals can derive for each formula through trial and error. Especially, it is difficult to derive span program for a formula with many input bits because number of elements of its matrix will increase exponentially. This paper proposes a method for optimal span program derivation, which formulates the problem as an optimization problem and solves it by evolutionary computation.

Keywords: span-program-based quantum algorithm, span program, evolutionary computation, Boolean fomula evaluation, quantum query complexity

1. はじめに

近年，与えられた論理式をできる限り少ない問合せ回数で正しく評価できる量子アルゴリズムが提案されている．量子クエリ計算量のモデルでは，関数 $f : \{0,1\}^n \to \{0,1\}$ をオラクルとして考え，各入力パターンをオラクルに問い合わせることによって出力を入手し，可能な限り少ない問合せ回数で関数の特性を決定する．オラクルへの問合せ回数のみを計算量として考え，コンピュータ内部の計算量は考えない．上記の研究は，2007 年に Farhi らが完全二分 NAND 木を $O(\sqrt{N})$ 量子時間で評価する量子アルゴリズムを考案したことにより発展した [1]．現在のコンピュータによる完全二分 NAND 木の評価は，乱択アルゴリズムによる問合せ計算量 $O(N^{.753})$ が最適であった [2], [3], [4]．NAND は

[1] 鹿児島大学大学院理工学研究科情報生体システム工学専攻
Department of Information Science and Biomedical Engineering, Graduate School of Science and Engineering, Kagoshima University, Korimoto, Kagoshima 890–0065, Japan
[a)] k4453895@kadai.jp
[b)] k4391399@kadai.jp

表 1　最適な Span Program が既知の論理式と量子対向界の例 [11]
Table 1　Example Boolean expressions whose optimal span program are known and their quantum adversary bounds [11].

Gate f	$Adv^{\pm}(f)$
$x_1 \wedge x_2$	$\sqrt{2}$
$x_1 \vee x_2$	$\sqrt{2}$
$x_1 \oplus x_2$	2
$Maj_3\left(x_{[3]}\right) = (x_1 \wedge x_2 \vee ((x_1 \vee x_2) \wedge x_3))$	2
$x_1 \vee x_2 \vee x_3$	$\sqrt{3}$
$x_1 \oplus x_2 \oplus x_3$	3
$x_1 \vee x_2 \vee x_3 \vee x_4$	2

古典コンピュータにおける万能論理ゲートであり，NAND で構成される論理式を効率良く評価できることは，多くの問題の計算向上につながる．そのため，NAND で構成される論理式（NAND 木）をもとに量子アルゴリズムが考案されていた．当初，完全二分という制約があったため，限定的な状況でしか使えなかったが，Ambainis らによって任意の NAND 木を評価できるようになった [5], [6], [7].

Reichardt らによって，Span Program をもとにした論理式評価の量子アルゴリズム（Span-Program-based Quantum Algorithm: SPQA）が提唱された [8]．これにより，論理式の評価対象は NAND 木だけでなく，論理式全般に拡大した．さらに，Reichardt は任意の論理式において，SPQA の量子問合せ計算量の下限と一般量子対向界（General Adversary Bound）が一致することを示した [9]．これは，論理式評価に Span Program を用いることで，最適な SPQA の導出が可能であることを表している．

最適な Span Program の導出には線形代数の式変換など数学的な方法が一般的である．入力ビット数が大きな論理式では行列の要素数が指数関数的に増加するため，数学的な導出は困難である．また，最適な Span Program が既知である複数の論理式を合成することで Span Program を導出する方法が提案されており [8]，一部の論理式でのみ，最適な Span Program が導出されている [10]．現在，最適な Span Program が見つかっている論理式は，入力が 3 bit 以下と 4 bit の一部のみである [11]（**表 1**）.

本研究では，最適な Span Program の導出を最適化問題として定式化し，進化計算を用いて最適な Span Program の近似解を導出する手法を提案する．

2. 研究分野の概要

2.1 量子ビット

量子ビットは量子コンピュータにおける情報の基本単位である．古典ビットが 0 か 1 のどちらか一方のみを格納するのに対し，量子力学では，重ね合わせの原理が成り立つため，量子ビットは 0 と 1 を重ね合わせ状態で格納する．式 (1) は 1 量子ビットの量子状態を表したものである．

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \qquad \left(|\alpha|^2 + |\beta|^2 = 1\right) \tag{1}$$

各項にかかる係数 α，β は確率振幅と呼ばれる複素数である．量子状態はいかなる操作を受けても，正規化の条件である $\langle\psi|\psi\rangle = 1$ を満たしている必要がある．量子ビットは観測されると重ね合わせ状態が壊れ，確率的に 0 か 1 のどちらかに収束する．

量子状態はヒルベルト空間 $\mathcal{H}$ 上における列ベクトルとして数学的に記述することができる．ここで，ヒルベルト空間 $\mathcal{H}$ とは内積が定義された複素ベクトル空間を指す．量子力学において，ヒルベルト空間は一般的に無限次元であるが，量子コンピュータでは有限次元として考えることができる．式 (1) を数学的に記述した場合，式 (2) のように書ける．

$$|\psi\rangle = \alpha \begin{bmatrix} 1 \\ 0 \end{bmatrix} + \beta \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha \\ \beta \end{bmatrix} \tag{2}$$

多量子ビットの状態ベクトルは，2 次元ヒルベルト空間のテンソル積 $\otimes$ により決定され，n 量子ビットのとき，2^n 次元ヒルベルト空間の列ベクトルになる．多量子ビットの表記方法は様々で，以下に示すものはすべて等価である．

$$|v\rangle \otimes |w\rangle = |v\rangle\,|w\rangle = |v, w\rangle = |vw\rangle \tag{3}$$

2.2 Span Program に基づく量子アルゴリズム（SPQA）

2.2.1 Span Program

Span Program は，Karchmer らによって提唱された論理式を評価する線形代数モデルである [12]．古典的な計算複雑性理論の分野で下界の証明 [12], [13] や Monotone Span Program として暗号処理の秘密分散などに用いられる [14].

n bit 入力の論理式に対する Span Program P は，複素ベクトル空間 $\mathbb{C}$ 上のターゲットベクトル $|t\rangle$（$\neq \vec{0}$）と，入力ベクトルの集合 $\{|v_k\rangle : k \in K\}$ から構成される．ここで，K は 1 から入力ベクトルの総数までの自然数からなる集合である．各入力ベクトル $|v_k\rangle$ には，入力ラベル I_k をラベル付けする．入力ラベル I_k は，論理式の入力 x を構成する各論理変数 $x_1, x_2, \ldots, x_n$ とその否定からなる論理積である．各入力ベクトル $|v_k\rangle$ に付けられた入力ラベル I_k が $true$ のとき，その $|v_k\rangle$ が選択され，論理式評価に用いられる．

以下では，Span Program を用いた論理式の評価方法について述べる．Span Program P がある論理式 $f_P : \{0,1\}^n \to \{0,1\}$ を評価するとき，式 (4) が成り立つ．

$$f_P(x) = \begin{cases} 1 & \text{if} \quad |t\rangle \in Span\{|v_k\rangle : I_k = true\} \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

すなわち，論理式に入力 x を与える際，入力ラベル $I_k = true$ となる $|v_k\rangle$ の線形結合で $|t\rangle$ を表すことができる場合は論理式 f_P の出力が 1 となる．$|t\rangle$ を線形結合で表

表 2　Maj_3 の Span Program の出力と真理値表
Table 2　Outputs and a truth table of Maj_3 span program.

x_1	x_2	x_3	線形結合	$Maj_3(x_1,x_2,x_3)$
0	0	0	not lie	0
0	0	1	not lie	0
0	1	0	not lie	0
0	1	1	$\frac{1}{2+\omega}\|v_2\rangle + \frac{1+\omega}{2+\omega}\|v_3\rangle$	1
1	0	0	not lie	0
1	0	1	$\frac{1+\omega}{2+\omega}\|v_1\rangle + \frac{1}{2+\omega}\|v_3\rangle$	1
1	1	0	$\frac{1}{2+\omega}\|v_1\rangle + \frac{1+\omega}{2+\omega}\|v_2\rangle$	1
1	1	1	$\frac{1}{3}\|v_1\rangle + \frac{1}{3}\|v_2\rangle + \frac{1}{3}\|v_3\rangle$	1

すことができない場合，出力は 0 となる．

3 bit の Majority 関数 $Maj_3(x_1,x_2,x_3)$ を例に説明する．Maj_3 の Span Program は式 (5) のように，ターゲットベクトル $|t\rangle$，入力ベクトル $|v_1\rangle$，$|v_2\rangle$，$|v_3\rangle$ を要素として持つ行列 V_K，および，入力ラベル I_k を要素とする集合 $\mathscr{I}_K$ により表すことができる．

$$\mathscr{I}_K = \left\{ \begin{array}{ccc} x_1, & x_2, & x_3 \end{array} \right\}$$
$$|t\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix},\ V_K = \begin{pmatrix} \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} \\ 1 & \omega & \omega^2 \end{pmatrix} \tag{5}$$

ここで，$\omega = e^{2\pi i/3}$ である．行列 V_K の各列は，入力ベクトル $|v_1\rangle$，$|v_2\rangle$，$|v_3\rangle$ となる．すなわち，$|v_1\rangle = \left(\frac{1}{\sqrt{3}}, 1\right)^T$ になる．また，$|v_1\rangle$，$|v_2\rangle$，$|v_3\rangle$ には，それぞれ $I_1 = x_1$，$I_2 = x_2$，$I_3 = x_3$ がラベル付けされる．

Maj_3 の場合は，$|t\rangle$ を入力ベクトルの線形結合で表すために，$\{|v_k\rangle\}$ のうち少なくとも 2 つを用いる必要がある．Maj_3 に対する入力 $x = 011$ の例を以下に示す．$x_2 = 1$，$x_3 = 1$ であることから，$I_2 = I_3 = true$ となるので，選択される入力ベクトルは $|v_2\rangle$，$|v_3\rangle$ の 2 つである．式 (6) に示すように，$|t\rangle$ を $|v_2\rangle$ および $|v_3\rangle$ の線形結合で表現できるため，Maj_3 の出力は 1 となる．

$$|t\rangle = \frac{1}{2+\omega}|v_2\rangle + \frac{1+\omega}{2+\omega}|v_3\rangle \tag{6}$$

2.2.2　SPQA の概要

SPQA は，Span Program をもとに作られるグラフ構造を量子ウォークで解くことにより，論理式評価を行う量子アルゴリズムである．量子ウォークは，ランダムウォークの量子版であり，ここでは離散時間モデルを用いる．Reichardt らは，Span Program を隣接行列に持つ完全 2 部グラフを，量子ウォークで解くことで論理式評価ができることを示している [8]．

以下では，Reichardt らが提案したグラフ作成の方法を述べるために，Span Program を再定義する．任意の論理式を計算する Span Program P について，ベクトル空間 $\mathbb{C}$ 上におけるターゲットベクトルを $|t\rangle$，入力ベクトル $|v_i\rangle$ を各列の要素として持つ行列を A とし，入力ベクトルに付される入力ラベル $I_{j,b}$ の集合を $\mathscr{I}$ とする．また，入力値 x による $|v_i\rangle$ の選択を数学的に表現するため，射影行列 $\Pi(x)$，選択ラベル集合 $\mathscr{I}(x)$ を導入する．それぞれの定義を以下に示す．

$$A = \sum_{i\in\mathscr{I}} |v_i\rangle\langle i| \in \mathcal{L}\left(\mathbb{C}^{|\mathscr{I}|}, \mathbb{C}^l\right) \tag{7}$$
$$\mathscr{I} = \bigcup_{j\in[n], b\in\{0,1\}} I_{j,b} \tag{8}$$
$$\Pi(x) = \sum_{i\in\mathscr{I}(x)} |i\rangle\langle i| \in \mathcal{L}\left(\mathbb{C}^{|\mathscr{I}|}\right) \tag{9}$$
$$\mathscr{I}(x) = \bigcup_{j\in[n]} I_{j,x_j} \tag{10}$$

ここで，l は $|v_i\rangle$ の次元数を表し，$[n] = \{1,2,\ldots,n\}$，$I_{j,0} = \overline{x_j}$，$I_{j,1} = x_j$ である．

上記の定義と 2.2.1 項における定義との違いを述べる．V_K は A の不要な部分を除いた行列である．入力ラベル $I_{j,b}$ は 1 つの論理変数またはその否定であり，2 つ以上の論理変数（およびその否定）の積を含まない点で 2.2.1 項の I_k とは異なる．入力ベクトル $|v_i\rangle$ は，2.2.1 項の添字 k は数値であるのに対して，論理積 $i = I_{j,x_j}$ を添字としている．射影行列 $\Pi(x)$ は，2.2.1 項の定義には表れないが，A の各列ベクトルのうち入力 x により選択される列のみを残すことで，入力ベクトル $\{|v_i\rangle\}$ の選択を数学的に表現するために定義する．すなわち，$\Pi(x)$ は行列 A に右から作用することにより，A の $i \in \mathscr{I}(x)$ がラベル付けされた列を保持し，他の列の要素をすべて 0 にする．

以上のように P を定義すると，SPQA で用いる重み付き完全 2 部グラフの隣接行列 $B_{G_P(x)}$ を次式のように表すことができる [15]．

$$B_{G_P(x)} = \begin{pmatrix} |t\rangle & A \\ O & E - \Pi(x) \end{pmatrix} \tag{11}$$

ここで，E は単位行列である．$B_{G_P(x)}$ とグラフ構造の対応を**図 1** に示す．A_{OJ} は A の 1 行目，A_{CJ} は A の 2 行目以降である．また，$A_{IJ} = E - \Pi(x)$ である．a_O，a_J，b_O，b_C，b_I は図 1 のように $B_{G_P(x)}$ にラベルとして配置され，グラフ構造ではノードに対応する．$B_{G_P(x)}$ から各ノード間の枝に重み付けし，グラフ構造を作成する．重みが 0 であれば，ノード間の枝はないものと見なす．例として，Maj_3 の Span Program を式 (5) としたときの $B_{G_P(000)}$ の隣接行列を式 (12) に，グラフを**図 2** に示す．

$$B_{G_P(000)} = \left(\begin{array}{c|cccccc} 1 & \frac{1}{\sqrt{3}} & 0 & \frac{1}{\sqrt{3}} & 0 & \frac{1}{\sqrt{3}} & 0 \\ 0 & 1 & 0 & \omega & 0 & \omega^2 & 0 \\ \hline 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right) \tag{12}$$

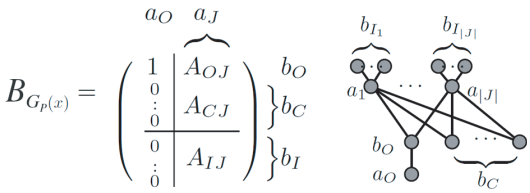

図 1　隣接行列 $B_{G_P(x)}$ によるグラフ構造の作成方法
Fig. 1　Adjacency matrix $B_{G_P(x)}$ and its graph.

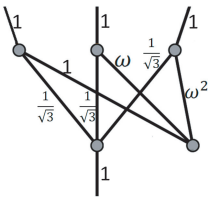

図 2　Maj_3 におけるグラフ構造
Fig. 2　Graph of Maj_3.

2.3　量子クエリ計算量

量子アルゴリズムでは問合せ回数を量子クエリ計算量（Quantum Query Complexity）と呼ぶ．量子ウォークで解く際，問合せ回数が適切でなければ，正しい結果を導くことができない．そのため，正しい結果が導きやすく，かつ，少ない問合せ回数で実行することが望まれる．SPQA の量子クエリ計算量は Span Program から導出される witness size に等しいことが示されている [10]．したがって，witness size の少ない Span Program を求めることで最適な SPQA を導出することができる．

2.3.1　witness size

witness size は Span Program の性能評価の指標として用いられる [9]．任意の論理式 f を計算する Span Program P を適用させた SPQA の量子クエリ計算量を $Q(f)$ とおくと，次式が成り立つ．

$$Q(f) = O(wsize(P)) \tag{13}$$

Span Program P の witness size $wsize(P)$ は次式より求める．

$$wsize(P) = \max_{x\in\{0,1\}^n} wsize(P,x) \tag{14}$$

$f_P(x) = 1$ であれば，入力ベクトルの線形結合によりターゲットベクトル $|t\rangle$ を表すことができるので，これを $|t\rangle \in Range(A\Pi(x))$ と表現する[*1]．このとき，$A\Pi(x)|w\rangle = |t\rangle$ を満たす $|w\rangle \in \mathbb{C}^{|\mathscr{J}|}$ が存在し，$wsize(P,x)$ を式 (15) のように表すことができる．ここで，$\mathbb{C}^{|\mathscr{J}|}$ は $|\mathscr{J}|$ 次元の複素空間を表す．

$$wsize(P,x) = \min_{|w\rangle : A\Pi(x)|w\rangle = |t\rangle} ||\,|w\rangle\,||^2 \tag{15}$$

[*1] 式 (4) の *Span* と *Range* はおよそ同義であるが，$A\Pi(x)$ のように行列であるか $|v_k\rangle$ のように列ベクトルであるかにより表現方法が異なる．

$f_P(x) = 0$ のとき，$|t\rangle \notin Range(A\Pi(x))$ となる．このとき，$\langle t|w'\rangle = 1$ かつ $\Pi(x)A^\dagger|w'\rangle = 0$ を満たす $|w'\rangle$ が存在し，$wsize(P,x)$ は以下のように表すことができる．

$$wsize(P,x) = \min_{\substack{|w'\rangle : \langle t|w'\rangle = 1 \\ \Pi(x)A^\dagger|w'\rangle = 0}} ||A^\dagger|w'\rangle||^2 \tag{16}$$

ここで，$A^\dagger$ は，A の転置複素共役を表す．

2.3.2　SPQA の最適性

量子クエリ計算量や論理式複雑計算量の下界の証明は，数理計画問題の分野として研究が進められている．量子クエリ計算量の下界指標には現在，多項式法（Polynomial Method）[16]，量子対向界（Adversary Bound）[17], [18]，一般量子対向界（General Adversary Bound, Negative Adversary Bound）[19] の 3 通りがある [20]．

Reichardt は任意の論理式 $f : \{0,1\}^n \to \{0,1\}$ において，Span Program P が f を計算するとき，SPQA の量子クエリ計算量の下限と一般量子対向界 $Adv^{\pm}(f)$ が一致することを示した [9]．これにより，一般量子対向界に等しい witness size を与える Span Program を求めることで，最適な SPQA の導出が可能である．

2.4　最適な Span Program の導出

Reichardt らによって定式化された最適な Span Program の導出について述べる [9]．一般量子対向界 $Adv^{\pm}(f)$ は次式により導出される．

$$Adv^{\pm}(f) = \min \max_{x\in\{0,1\}^n} \sum_{j\in[n]} \langle x|X_j|x\rangle \tag{17}$$

$$s.t. \quad \sum_{j:x_j\neq y_j} \langle x|X_j|y\rangle = 1 \text{ if } f(x) \neq f(y) \tag{18}$$

$$X_j \succeq 0 \tag{19}$$

ここで，$x = (x_1, x_2, \ldots, x_n)$，$y = (y_1, y_2, \ldots, y_n)$ は論理式の入力を表す．X_j は Span Program を間接的に構成する行列であり，式 (17)，(18)，(19) で表される最適化問題の設計変数となる．

X_j の集合から，Span Program を構成する行列 A を以下のように導出することができる．まず行列 X_j の集合 $\{X_1, X_2, \ldots, X_n\}$ を，式 (20) に従ってコレスキー分解することで，行列 A を構成する要素である $\{|u_{xj}\rangle\}$ を求める．

$$\{|u_{xj}\rangle\} : \langle u_{xj}|u_{yj}\rangle = \langle x|X_j|y\rangle \tag{20}$$

この $|u_{xj}\rangle$ を用いて，式 (21) によって行列 A を導出する．

$$A := \sum_{x\in\mathscr{F}_0, j\in[n]} |x\rangle\langle j, \overline{x_j}| \otimes \langle u_{xj}| \tag{21}$$

ここで，$\mathscr{F}_0 = \{x : f(x) = 0\}$ である．行列 A は，$x \in \mathscr{F}_0$ を入力した場合，選択される入力ベクトルのある行要素が共通してすべて 0 になるため，ターゲットベクトルを表せ

ないようになっている．

以上のように，行列 X_j の集合 $\{X_1, X_2, \ldots, X_n\}$ を求めることで，最適な Span Program を導出することができるものの，X_j の一般的な導出手法は明らかになっていない．

2.5 進化計算アルゴリズム

進化計算は多点探索であることを特徴とするメタヒューリスティクスであり，不連続，多峰性，ノイズを含むなどの特徴を持つ問題においても柔軟に探索を行える頑健な最適化手法である．なお，本節の数式で用いる変数や記号などは，進化計算分野で一般的に用いられる変数や記号などを用いる．このため本節でのみ，x，v などの変数が，本論文の他の章節と異なる意味で用いられる点に注意されたい．

2.5.1 差分進化（DE）

差分進化（Differential Evolution: DE）[21] は実数値最適化問題を対象とする進化計算手法の 1 つである．典型的な実数値 GA や進化戦略と比較し，DE は最適解への収束が早く頑健であることが示されている [22], [23].

DE は一般的に，DE/base/num/cross のように表記することで戦略の違いを表す．base はベースベクトルの選択方法を示し，最良の個体を用いる最良選択（best）や，個体群からランダムに選ぶランダム選択（rand）などがある．num は変異ベクトルを生成する際に差分を行う個体対の数を示し，cross は二項交叉（binomial crossover: bin）や指数交叉（exponential crossover: exp）などの交叉方法を示す．

ここでは DE/rand/1/bin について説明する．DE では，現世代の個体群に含まれるすべての個体が次の世代の個体の生成に関与する．世代 g の各個体群における i 番目の個体を $x_{i,g}$ と示す．各世代で，各個体（ターゲットベクトル）$x_{i,g}$ に対して，ベースベクトル $x_{b,g}$ と差分に用いる個体対 $x_{r1,g}$，$x_{r2,g}$ を個体群の中から個体が重ならないようにランダムに選択し，式 (22) を用いて変異ベクトル $v_{i,g}$ を生成する．

$$v_{i,g} = x_{b,g} + S(x_{r1,g} - x_{r2,g}) \tag{22}$$

ここで S をスケール係数と呼ぶ．

次に交叉を行い，ターゲットベクトルと変異ベクトルからトライアルベクトル $u_{i,g}$ を作成する．二項交叉では交叉率 CR（$0 \leq CR \leq 1$）とランダムに選択した添字 j_{rand}（$1 \leq j_{rand} \leq D$）（D は次元数）に基づき，個体中の要素 j を式 (23) のように決定する．

$$u_{i,j,g} = \begin{cases} v_{i,j,g} & \text{if } rand_{i,j}[0,1] \leq CR \text{ or } j = j_{rand} \\ x_{i,j,g} & \text{otherwise} \end{cases} \tag{23}$$

$rand_{i,j}[0,1]$ は範囲 $[0,1]$ の一様乱数である．

上記により生成された $u_{i,g}$ と $x_{i,g}$ のうち，目的関数の値が優れる方を次世代に残す．

DE において設定するパラメータは個体数，スケール係数 S，交叉率 CR の 3 つである．個体数は $5D$〜$10D$ 程度が推奨されているが，実際にはそれよりも少ない個体数で十分な探索性能を持つことが示されている [21]．スケール係数 S を小さくすると収束性が向上し，大きくすると多様性が向上する．交叉率 CR の特性は問題の評価関数に依存するため，分離可能問題に対しては $CR \in [0, 0.2]$，関数の景観が未知の場合は，変数間の依存を考慮して $CR \in [0.9, 1]$ と設定することが推奨されている [21], [24].

2.5.2 Generalized Opposition-based DE（GODE）

Generalized Opposition-based DE（GODE）[25], [26] は DE の操作に加えて，Generalized Opposition-based Learning（GOBL）という手法を導入することで，探索性能の向上を図った手法である．GOBL は計算知能の新しい手法であり，DE 以外の進化計算手法にも適用されている．主な操作としては，現在の個体群 P の探索空間に対して対称となる探索空間に対称個体群 GOP を生成し，P と GOP の個体群の中から上位の個体を個体数分だけ選択し，次の世代の個体群 P' とする．GOP を生成する手法として 4 種類のモデルが提案されており，本研究では最も良いモデルとされている Random-GOBL を用いる [25], [26]. Random-GOBL は以下の式によって，対称個体の生成を行う．

$$x^*_{i,j} = k(a_j + b_j) - x_{i,j} \tag{24}$$

$$a_j = \min(x_{i,j}),\ b_j = \max(x_{i,j}) \tag{25}$$

$$x^*_{i,j} = rand(a_j, b_j) \text{ if } x^*_{i,j} \notin [x_{min}, x_{max}] \tag{26}$$

$$i = 1, 2, ..., N_P,\ j = 1, 2, ..., D,\ k = rand(0, 1)$$

GOBL では個体 i の j 次元の値 $x_{i,j}$，生成する対称個体の j 次元の値 $x^*_{i,j}$，現在の個体群における j 次元の最小値 a_j と最大値 b_j，$[a_j, b_j]$ 内のランダムな値 $rand(a_j, b_j)$，設計変数の定義域 $[x_{min}, x_{max}]$，個体数 N_P，$[0, 1]$ のランダムな値 $rand(0, 1)$，世代ごとに設定される値 k を用いる．k の値がランダムに変化することで，世代ごとに生成される対称個体の位置が変化する．生成された対称個体が問題の探索空間の範囲外であった場合は，現在の探索空間内でランダムに生成する．

GODE は GOBL を行う確率 p_o を設定し，世代ごとに GOBL と DE の進化的操作を確率的に切り替える．また，初期個体群を生成する際にも GOBL を適用し，この際の最小値 a_j と最大値 b_j は，設計変数の定義域 x_{min} と x_{max} を用いる．対称個体を生成することにより，現在の探索空間から新しい探索空間を探索することが可能となり，より良い解を発見する機会を与えることができる．

2.5.3　分散共分散行列の適合に基づく進化戦略（CMA-ES）

CMA-ES [27] は，進化的戦略（Evolutionary Strategy: ES）の一種である．単峰性関数や設計変数間に依存関係がある問題に対して有効である [28]．

CMA-ES は多変量正規分布を用いた突然変異によって探索点を生成する．まず，個体を評価し，評価の高い個体情報から，突然変異分布の平均ベクトルと共分散行列を得る．次に，上記突然変異分布の平均ベクトルを探索範囲の中心とし，突然変異分布の形状と大きさを分散共分散行列から算出する．分布に従って生成した探索点集合の適応度に基づいてより優れた解が得られると予想される方向にパラメータを更新する．

3.　提案する方式

3.1　定式化

本論文では，古典的コンピュータの進化計算を用いて，最適な Span Program の近似解を導出する手法を提案する．定式化においては，2.4 節で述べた記述をもとに行う．すなわち，式 (18) のもとで目的関数 F を式 (27) のように定義し，F を最小化するような $\{X_1, X_2, \ldots, X_n\}$ を求める最適化問題として定式化する．設計変数は $\{X_1, X_2, \ldots, X_n\}$ の各要素で，制約条件は式 (18) とする．このとき，F の最小値は 0 である．

$$F = \begin{cases} \max_x \sum_{j\in[n]} \langle x| X_j |x\rangle - Adv^{\pm}(f) & \text{if } X_j \succeq 0 \\ P(X_j) & \text{otherwize} \end{cases} \tag{27}$$

ここで，$P(X_j)$ は制約違反に対するペナルティ関数である．

定式化の具体例として，$f(x) = x_1 \vee x_2$（OR_2）のときの目的関数 F の 1 式目と制約条件の 1 式目 (18) を，それぞれ式 (28)，式 (29) に示す．ただし，$X_{j,(k,l)}$ は X_j の要素 (k,l) のことを指す．X_j が対称行列（$X_{j,(k,l)} = X_{j,(l,k)}$）であるので，式 (29) は，$k \le l$ における制約条件を表す．

$$F = \max_{k\in\{00,01,10,11\}} \sum_{j\in\{1,2\}} X_{j,(k,k)} - \sqrt{2} \tag{28}$$

$$s.t. \quad \begin{cases} X_{1,(00,10)} = 1 \\ X_{2,(00,01)} = 1 \\ X_{1,(00,11)} + X_{2,(00,11)} = 1 \end{cases} \tag{29}$$

3.2　設計変数の削減

本方式では，$\{X_1, X_2, \ldots, X_n\}$ のすべての要素を設計変数とするのではなく，式 (18) と，式 (19) の一部により自動的に定まる要素については設計変数から除外する．制約は式 (18)（条件 A）および式 (19) を満たすことであり，後者は，$\{X_1, X_2, \ldots, X_n\}$ が対称行列（条件 B）かつ固有値がすべて 0 以上であること（条件 C）である．本手法では，条件 A および条件 B をつねに満たすように，式 (18) に従って設計変数を削減する．

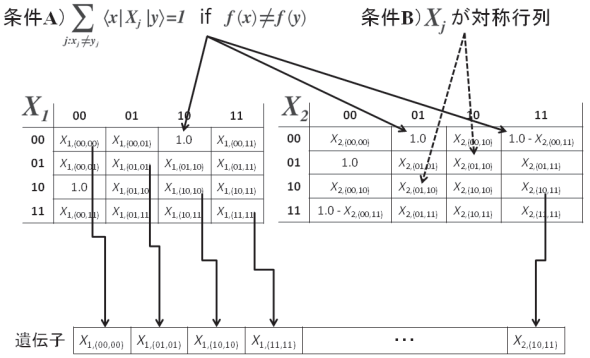

図 3　OR_2 における遺伝子表現
Fig. 3　Chromosome representation of OR_2.

例として，OR_2 を対象とする場合の設計変数について**図 3** および以下で述べる．OR_2 では $\{X_1, X_2\}$ の各要素が設計変数となる．このとき，条件 A（OR_2 の場合，式 (29)）より，式 (29) に示すように一部の成分が算出される．このため，式 (29) によって決定される 3 個の成分 $X_{1,(00,10)}$，$X_{2,(00,01)}$，$X_{2,(00,11)}$ を除いた 17 個の成分を設計変数とする．

上記のように設計変数を設けることで，条件 A および B はつねに満たされるため，最適化の過程では条件 C のみを考慮すればよい．すなわち，$X_1, X_2, \ldots, X_n$ の固有値のうち 0 または負となる固有値の個数を m として，以下のペナルティ関数を用いる．ここで，T_{Pen} は定数とする．

$$P(X_j) = m \times T_{Pen} \tag{30}$$

3.3　処理手順

本手法を用いて最適な Span Program を導出する手順を述べる．まず，対象となる論理式 $f(x)$ の一般量子対向界 $Adv^{\pm}(f)$ を導出する．本論文では，Reichardt らの論文を参考にした [8]．次に 3.2 節に従って設計変数を決定し，2.5 節に示した進化計算アルゴリズムを用いて最良個体を探索する．個体の適応度は式 (27) の目的関数 F より導出する．その後，得られた最良個体に対して 3.2 節の逆の操作を行うことにより，行列 $X_1, X_2, \ldots, X_n$ を生成する．2.4 節に示した手順により $X_1, X_2, \ldots, X_n$ から，Span Program を構成する行列 A を導出する．

行列 A が論理式 $f(x)$ を評価する Span Program であるかどうかは，2.2.1 項に示した手順で確認できる．また，A により構成される Span Program の最適性は，2.3.1 項に示す手順により witness size を導出し，一般量子対向界 $Adv^{\pm}(f)$ または既知の最適解の witness size と比較することで評価できる．

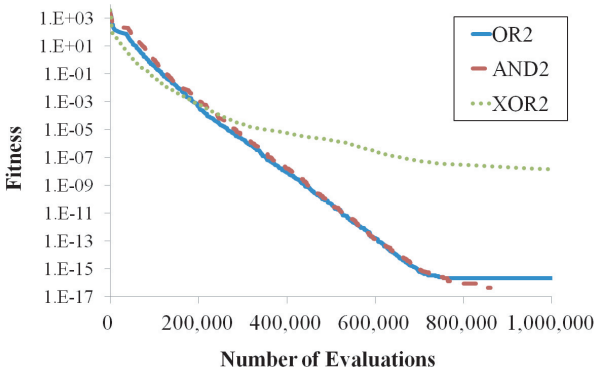

図 4　OR_2，AND_2，XOR_2 における適応度の推移グラフ
Fig. 4　Fitness transitions on OR_2, AND_2 and XOR_2.

4. 評価実験

4.1 概要

提案する方式により，最適な Span Program の近似解の導出を試みる実験を行った．4.2.1 項および 4.2.2 項では，それぞれ 2 bit，3 bit の論理式を対象として，DE を用いて実験を行った．4.2.3 項では 4 bit の論理式を対象として，GODE および CMA-ES を順次適用することで Span Program の導出を試みた．得られた近似解の評価は 3.3 節で述べた手順で行った．なお，以下では $T_{Pen} = 1{,}000$ として実験を行った．

4.2 実験結果

4.2.1 2 bit の論理式

まず 2 bit の論理式を対象として，進化計算により得られる近似解の品質の確認を行った．DE/rand/1/bin を用いて最適化を行うものとし，試行回数を 5 回とした．対象とした論理式は OR_2，AND_2 および XOR_2 で，それぞれの次元数は 17，17，16 である．集団サイズを 100 個体，終了条件を 10,000 世代，評価回数の上限を 1,000,000 回，交叉率 CR を 0.9，スケール係数 S を 0.5 とした．

OR_2，AND_2，XOR_2 における最良個体の適応度の推移を**図 4** に示す．図 4 に示す結果は，5 回の試行で得られた結果の平均である．図 4 より，いずれの論理関数においても評価回数が大きくなるにつれて適応度が収束したことが分かる．

次に，実験によって得られた Span Program を評価した．各実験を行った論理式の一般量子対向界（$Adv^{\pm}(f)$），既知の最適解の witness size，実験によって得られた Span Program の witness size を**表 3** に示す．表 3 に示した結果は，5 回の試行で得られた最良解のなかで，最も良い witness size を示している．表 3 より，いずれの論理式においても，$Adv^{\pm}(f)$ と非常に近い witness size を持つ Span Program を導出できたことが分かる．

表 3　実験を行った 2 bit の論理式における $Adv^{\pm}(f)$ と witness size
Table 3　$Adv^{\pm}(f)$ and witness sizes of the tested 2-bit Boolean expressions.

論理式名	$Adv^{\pm}(f)$	既知の最適解の witness size	得られた解の witness size
OR_2	$\sqrt{2}$	$\sqrt{2}$	1.41421
AND_2	$\sqrt{2}$	$\sqrt{2}$	1.41421
XOR_2	2	2	2.00000

注：$\sqrt{2} \approx 1.41421$

表 4　実験を行った論理式（3 bit）
Table 4　Tested 3-bit Boolean expressions.

論理式名	Gate	次元数
OR_3	$x_1 \vee x_2 \vee x_3$	101
AND_3	$x_1 \wedge x_2 \wedge x_3$	101
Maj_3	$x_1 \wedge x_2 \vee ((x_1 \vee x_2) \wedge x_3)$	92
$Parity_3$	$x_1 \oplus x_2 \oplus x_3$	92
If-Then-Else	$(x_2 \wedge x_3) \vee (x_1 \wedge \overline{x_3})$	92
$Equal_3$	$(x_1 \wedge x_2 \wedge x_3) \vee (\overline{x_1} \wedge \overline{x_2} \wedge \overline{x_3})$	96

表 5　実験を行った 3 bit の論理式における $Adv^{\pm}(f)$ と witness size
Table 5　$Adv^{\pm}(f)$ and witness sizes of the tested 3-bit Boolean expressions.

論理式名	$Adv^{\pm}(f)$	既知の最適解の witness size	得られた解の witness size
OR_3	$\sqrt{3}$	$\sqrt{3}$	1.73471
AND_3	$\sqrt{3}$	$\sqrt{3}$	1.73808
Maj_3	2	2	2.00921
$Parity_3$	3	3	3.02427
If-Then-Else	2	2	2.00921
$Equal_3$	$3/\sqrt{2}$	$3/\sqrt{2}$	2.16242

注：$\sqrt{3} \approx 1.73205$，$3/\sqrt{2} \approx 2.12132$

4.2.2 3 bit の論理式

表 4 に示す 3 bit の論理式を対象として，Span Program の導出を試みた．試行回数を 1 回とした．なお，DE/rand/1/bin を用いて予備実験を行ったところ収束が遅かったため，3 bit の論理式を対象とした実験では DE/rand/1/exp を用いた．集団サイズを 1,000 個体，世代数の上限を 50,000 世代とした．すなわち，評価回数の上限は 5×10^7 回となる．交叉率 CR を 0.9，スケール係数 S を 0.3 とした．

実験によって得られた Span Program を評価し，その witness size を**表 5** に示す．表 5 には，各実験を行った論理式の $Adv^{\pm}(f)$，既知の最適解の witness size をあわせて示す．表 5 より，いずれの論理式においても，小数第 1 位まで $Adv^{\pm}(f)$ と等しい witness size の Span Program を導出できたことが分かる．

4.2.3 4 bit の論理式

4 bit の論理式を対象とする場合は，次元数が膨大になるため，異なる進化計算アルゴリズムを 2 段階で適用することで Span Program の導出を試みた．

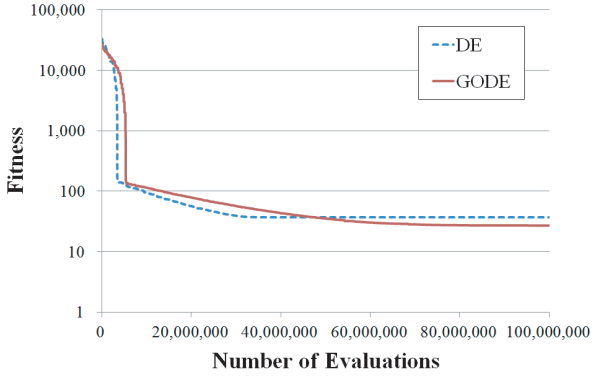

図 5　第 1 段階における適応度の推移（OR_4）
Fig. 5　Fitness transitions on the first search (OR_4).

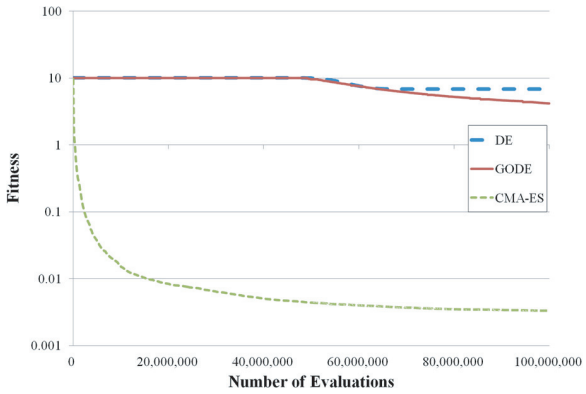

図 6　第 2 段階における適応度の推移（OR_4）
Fig. 6　Fitness transitions on the second search (OR_4).

まず，OR_4 を対象として予備実験を行い，使用するアルゴリズムの選定を行うこととした．第 1 段階では，ランダムに生成した初期個体を対象として DE，GODE で最適化を行った．第 2 段階では，得られた最良個体を初期集団に反映させて再度，DE，GODE，CMA-ES で最適化を試みた．すなわち，DE，GODE では初期集団に得られた最良個体を加え，CMA-ES では初期集団生成に用いる平均ベクトルを得られた最良個体とした．DE は 3 bit の論理式の実験と同様，DE/rand/1/exp を用い，集団サイズを 1,000 個体，$CR = 0.9$，$S = 0.2$ とした．GODE は DE と同様のパラメータに加え，ジャンプ率 p_o を文献 [25] に従って 0.4 とした．CMA-ES は文献 [29] を参考に，集団サイズを 22 個体，$\sigma = 0.02$ とした．

第 1 段階における，各アルゴリズムにより発見された最良個体の適応度の推移を**図 5** に示す．図 5 より，1×10^7 回以内の評価回数で，DE，GODE の両方ともすべての制約条件にあった個体（適応度が 1,000 未満の個体）の発見に成功したものの，十分に小さい witness size を持つ解を発見できなかった．

第 1 段階では GODE の方が最終的に良い解を発見できたため，第 2 段階の最適化を行う際に，第 1 段階の最終世代における GODE の最良個体を新しい初期個体の 1 つとして加え，DE，GODE，CMA-ES を用いて最適化を試みた．GODE の最良個体を初期個体として与えた際の，OR_4 における適応度の推移グラフを**図 6** に示す．図 6 より，DE，GODE に関しては第 2 段階開始後，再度大域的な探索を行ったため探索の集中化が進む 6.0×10^7 回程度までは最良解の更新を行えず，十分な品質の近似解は発見できなかった．これに対し，CMA-ES においては局所解に陥ることなく探索を継続できたことが分かる．

上記の予備実験の結果を受けて，4 bit の論理式を対象とした実験では，まず GODE を適用し，GODE で得られた最良個体を初期個体として CMA-ES で最適化を行うこととした．それぞれの評価回数の上限は図 6 を参考に，第 1 段階の GODE は 3×10^8 回，第 2 段階の CMA-ES は 2.2×10^7 回とした．実験を行った 4 bit の論理式を**表 6** に示す．試行回数は 1 回とした．

表 6　実験を行った論理式（4 bit）
Table 6　Tested 4-bit Boolean expressions.

論理式名	Gate	次元数	最適解が既知か
OR_4	$x_1 \vee x_2 \vee x_3 \vee x_4$	529	Yes
AND_4	$x_1 \wedge x_2 \wedge x_3 \wedge x_4$	529	Yes
$Parity_4$	$x_1 \oplus x_2 \oplus x_3 \oplus x_4$	480	Yes
Function #282	$(x_1 \vee x_3 \vee x_4) \wedge (\overline{x_3} \vee \overline{x_4}) \wedge ((\overline{x_1} \wedge \overline{x_2}) \vee (\overline{x_3} \wedge \overline{x_4}))$	496	No

表 7　実験を行った 4 bit の論理式における $Adv^{\pm}(f)$ と witness size
Table 7　$Adv^{\pm}(f)$ and witness sizes of the tested 4-bit Boolean expressions.

論理式名	$Adv^{\pm}(f)$	既知の最適解の witness size	得られた解の witness size
OR_4	2	2	2.00181
AND_4	2	2	2.00513
$Parity_4$	4	4	4.03763
Function #282	2.80369	2.92535	2.84124

実験によって得られた Span Program の witness size を，各実験を行った論理式の $Adv^{\pm}(f)$，既知の最適解の witness size とともに**表 7** に示す．表 7 より，すべての論理式において，小数第 1 位まで $Adv^{\pm}(f)$ と等しい witness size を持つ Span Program を導出できたことが分かる．特に，最適解が見つかっていない *Function #282* に関しては，既知の最適解よりも witness size が小さい Span Program を求めることに成功した．

4.3　考察

実験に用いたすべての論理式において，最適な Span Program の近似解を導出できたことで，本手法が汎用的に用いることができると考える．特に，*Function #282* においても従来よりも最適な解を求めることができたことにより，最適解が未知の他の論理式においても，近似解の導出

が期待できる．

なお，3.1 節で述べた定式化では，最適な Span Program の導出における設計変数の総数は，n bit の論理式の場合に $n4^n$ となる．3.2 節で述べた設計変数の削減により，およそ 1/2 程度の設計変数の総数に抑えることができるものの，n が増えるにつれて次元数が指数関数的に増大する．このため，現段階において本方式は，たかだか数個の論理変数を入力とする論理式を対象とした SPQA の設計に対して有効である．より多くの設計変数を含む SPQA を対象とするには，さらなる次元削減や，問題を分割して解くなどの工夫が必要である．

5. おわりに

SPQA で用いる最適な Span Program を，進化計算を用いて近似的に導出する方式を提案した．提案した方式は，従来は専門家が試行錯誤的に導出していた最適な Span Program を最適化問題として定式化し，大規模かつ複雑な適応度景観を持つ問題でも最適化を行える GODE や CMA-ES を用いて近似解を得る点に特徴がある．

実験により，提案する方式が，一般量子対向界の値に近い witness size を持つ Span Program を導出できることを示した．特に，最適解が未知の *Function #282* に関しては，既知のものよりも witness size が小さい Span Program を導出することができた．

今後は，最適解が未知の他の論理式に対する Span Program の導出や，5 bit 以上の論理式に対する Span Program の導出，および，近似解を用いた最適解発見の支援の具体的な方法を検討する．

謝辞　プログラムの実装，実験にご協力いただいた南和成氏に深く感謝する．

参考文献

[1] Farhi, E., Goldstone, J. and Gutmann, S.: A Quantum Algorithm for the Hamiltonian NAND Tree, *Theory of Computing*, Vol.4, No.8, pp.169–190 (2008).

[2] Snir, M.: Lower bounds on probabilistic linear decision trees, *Theoretical Computer Science*, Vol.38, pp.69–82 (1985).

[3] Saks, M. and Wigderson, A.: Probabilistic Boolean decision trees and the complexity of evaluating game trees, *27th Annual Symposium on Foundations of Computer Science*, pp.29–38, IEEE (1986).

[4] Santha, M.: On the Monte carlo boolean decision tree complexity of read-once formulae, *Random Structures & Algorithms*, Vol.6, No.1, pp.75–87 (1995).

[5] Childs, A.M., Reichardt, B.W., Špalek cur, R. and Zhang, S.: Every NAND formula on N variables can be evaluated in time $O(N^{\frac{1}{2}+\varepsilon})$, *arXiv preprint quant-ph/0703015*, pp.1–14 (2007).

[6] Ambainis, A.: A nearly optimal discrete query quantum algorithm for evaluating NAND formulas, *arXiv preprint arXiv:0704.3628*, pp.1–21 (2007).

[7] Ambainis, A., Childs, A.M., Reichardt, B.W., Špalek, R. and Zhang, S.: Any AND-OR Formula of Size N Can Be Evaluated in Time N 1/2+o(1) on a Quantum Computer, *SIAM Journal on Computing*, Vol.39, No.6, pp.2513–2530 (2010).

[8] Reichardt, B.W. and Špalek, R.: Span-program-based quantum algorithm for evaluating formulas, *Proc. 40th annual ACM symposium on Theory of computing*, pp.103–112, ACM (2008).

[9] Reichardt, B.W.: Span programs and quantum query complexity: The general adversary bound is nearly tight for every boolean function, *50th Annual IEEE Symposium on Foundations of Computer Science, 2009, FOCS'09*, pp.544–551, IEEE (2009).

[10] Reichardt, B.W.: Least span program witness size equals the general adversary lower bound on quantum query complexity, *Electronic Colloquium on Computational Complexity, Report*, No.75, pp.1–18 (2010).

[11] Reichardt, B.W. and Špalek, R.: Quantum query complexity of up to four-bit functions, available from ⟨http://www.ucw.cz/˜robert/papers/gadgets⟩.

[12] Karchmer, M. and Wigderson, A.: On Span Programs, *Structure in Complexity Theory Conference, 1993, Proc. 8th Annual*, pp.102–111, IEEE (1993).

[13] Babai, L., Gál, A. and Wigderson, A.: Superpolynomial lower bounds for monotone span programs, *Combinatorica*, Vol.19, No.3, pp.301–319 (1999).

[14] Beimel, A., Gál, A. and Paterson, M.: Lower bounds for monotone span programs, *Computational Complexity*, Vol.6, No.1, pp.29–45 (1996).

[15] Reichardt, B.W.: Span programs and quantum query algorithms, *Electronic Colloquium on Computational Complexity* (*ECCC*), Vol.17, p.110 (2010).

[16] Beals, R., Buhrman, H., Cleve, R., Mosca, M. and De Wolf, R.: Quantum lower bounds by polynomials, *J. ACM*, Vol.48, No.4, pp.778–797 (2001).

[17] Barnum, H.: Quantum query complexity and semidefinite programming, *Proc. 18th IEEE Annual Conference on Computational Complexity*, pp.179–193 (2003).

[18] Laplante, S., Lee, T. and Szegedy, M.: The quantum adversary method and classical formula size lower bounds, *Computational Complexity*, Vol.15, No.2, pp.163–196 (2006).

[19] Hoyer, P., Lee, T. and Špalek, R.: Negative weights make adversaries stronger, *Proc. 39th annual ACM symposium on Theory of computing*, pp.526–535, ACM (2007).

[20] 福原秀明：ブール関数の複雑さに関する研究，博士論文，東北大学 (2010).

[21] Storn, R. and Price, K.: Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces, *Journal of global optimization*, Vol.11, No.4, pp.341–359 (1997).

[22] Price, K., Storn, R.M. and Lampinen, J.A.: *Differential Evolution: A Practical Approach to Global Optimization* (*Natural Computing Series*), Springer-Verlag New York, Inc., Secaucus, NJ, USA (2005).

[23] Das, S. and Suganthan, P.: Differential Evolution: A Survey of the State-of-the-Art, *IEEE Trans. Evolutionary Computation*, Vol.15, No.1, pp.4–31 (2011).

[24] Ronkkonen, J., Kukkonen, S. and Price, K.: Real-parameter optimization with differential evolution, *The 2005 IEEE Congress on Evolutionary Computation, 2005*, Vol.1, pp.506–513 (2005).

[25] Wang, H., Wu, Z., Rahnamayan, S. and Kang, L.: A Scalability Test for Accelerated DE Using Generalized Opposition-Based Learning, *Proc. 2009 9th Interna-*

tional Conference on Intelligent Systems Design and Applications, ISDA '09, pp.1090–1095, IEEE Computer Society (2009).
[26] Wang, H., Wu, Z. and Rahnamayan, S.: Enhanced opposition-based differential evolution for solving high-dimensional continuous optimization problems, *Soft Comput.*, Vol.15, No.11, pp.2127–2140 (2011).
[27] Hansen, N. and Ostermeier, A.: Adapting arbitrary normal mutation distributions in evolution strategies: the covariance matrix adaptation, *Proc. IEEE International Conference on Evolutionary Computation, 1996*, pp.312–317 (1996).
[28] Hansen, N.: Invariance, Self-Adaptation and Correlated Mutations and Evolution Strategies, *Proc. 6th International Conference on Parallel Problem Solving from Nature, PPSN VI*, pp.355–364, Springer-Verlag (2000).
[29] Hansen, N.: Benchmarking a BI-population CMA-ES on the BBOB-2009 function testbed, *Proc. 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference: Late Breaking Papers, GECCO '09*, pp.2389–2396, ACM (2009).

佐多 恵悟

1988 年生．2013 年鹿児島大学工学部情報工学科卒業．現在，同大学大学院理工学研究科情報生体システム工学専攻所属．量子アルゴリズム設計に関する研究に従事．ニューラルネットワークとその応用の研究に興味．

松山 開

1991 年生．2013 年鹿児島大学工学部情報生体システム工学科卒業．現在，同大学大学院理工学研究科情報生体システム工学専攻所属．量子アルゴリズム設計に関する研究に従事．系列ラベリングに興味．

坂口 裕一

1989 年生．2012 年鹿児島大学工学部情報工学科卒業．現在，同大学大学院理工学研究科情報生体システム工学専攻所属．進化計算とその応用の研究に従事．

中山 茂　（正会員）

1977 年京都大学大学院博士課程修了，同年上智大学助手，1981 年京都工芸繊維大学助手，1987 年兵庫教育大学助教授，1997 年鹿児島大学工学部情報工学科教授．現在，同大学大学院情報生体システム工学専攻教授，京都大学工学博士．1996 年情報文化学会学会賞受賞．2000 年九州工学教育協会賞受賞．主として，量子情報工学，群知能，分散オブジェクト，進化的アルゴリズムの研究に従事．システム制御情報学会，電気学会，日本工学教育協会各会員．

小野 智司　（正会員）

2002 年筑波大学大学院博士課程工学研究科修了．2001 年日本学術振興会特別研究員．2003 年鹿児島大学工学部情報工学科助手．2010 年同大学大学院理工学研究科情報生体システム工学専攻准教授，現在に至る．博士（工学）．進化計算とその応用の研究に従事．2009 年人工知能学会研究会優秀賞，2013 年情報処理学会山下記念研究賞等受賞．IEEE，電子情報通信学会，人工知能学会，進化計算学会等各会員．