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]

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 の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，対象となる論理式毎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 Matuyama[†1,b)] Hirokazu Sakaguchi[†1] Shigeru Nakayama[†1] Satoshi Ono[†1]

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, 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い合わ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，入力の各ビット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い合わせ回数で関数の特性を決定する．オラクルへの問い合わ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い合わせ計算量 $O(N^{.753})$ が最適であった [2–4]．NAND は

†1 現在，鹿児島大学理工学研究科情報生体システム工学専攻
Presently with Department of Information Science and Biomedical Engineering, Graduate School of Science and Engineering, Kagoshima University,
Presently with 1-21-40, 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]

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 [5–7].

Reichardt とŠpalek 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論理式全般に拡張された．さらに，Reichardt は任意の論理式において，SPQA の量子問い合わ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 bit 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 が既知である論理式の複数合成により 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 3bit 以下と 4bit の一部のみである（表 1，[11] 参照）.

本研究では，進化計算を用いて最適な Span Program を導出する手法を提案する．現在までに最適な Span Program を発見できていない論理式について近似解を発見することで，最適な Span Program の導出を支援する．

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．

2.2 Span Program に基づく量子アルゴリズム

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，ベクトル空間 V 上のターゲットベクトル $|t\rangle(\neq \vec{0})$ と，入力ベクトル群 $\{|v_j\rangle : j \in J\}$ から構成される．入力ベクトル $|v_j\rangle$ には，論理式に対する入力 x の各 bit 集合 $\{x_1, \overline{x_1}, \ldots, x_n, \overline{x_n}\}$ の要素から作成できる論理積 I_j が各要素にラベル付けされる．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 (3) が成り立つ．

$$f_P(x) = \begin{cases} 1 & if\ |t\rangle \in Span\{|v_j\rangle : I_j = true\} \\ 0 & otherwise \end{cases} \tag{3}$$

すなわち，論理式に入力 x を与える際，論理積 $I_j = true$ となる $|v_j\rangle$ の線形結合で $|t\rangle$ を表すことができることと，論理式 f_P の出力が 1 となることは同義である．線形結合で $|t\rangle$ を表すことができない場合，出力は 0 となる．

3bit の Majority 関数 $Maj_3(x_1, x_2, x_3)$ を例に説明する．3bit の Majority 関数の Span Program は式 (4) のように，ターゲットベクトル $|t\rangle$ と，入力ベクトル群を要素として持つ行列 V_J により表すことができる．

$$I_J = \left\{ \begin{matrix} x_1 & x_2 & x_3 \end{matrix} \right\}$$
$$|t\rangle = \begin{pmatrix}1\\0\end{pmatrix},\ V_J = \begin{pmatrix} \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} \\ 1 & \omega & \omega^2 \end{pmatrix} \tag{4}$$

ここで，$\omega = e^{2\pi i/3}$ である．行列 V_J の各列は，入力ベクト

表 2 Maj_3 の Span Program の出力と真理値表

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

ル群 $\{|v_1\rangle, |v_2\rangle, |v_3\rangle\}$ となる．例えば Maj_3 の場合，$|v_1\rangle$ は $\left(\frac{1}{\sqrt{3}}, 1\right)^T$ になる．I_J は論理積 I_j の群であり，$I_1, I_2, \ldots$ は V_J の列の $|v_1\rangle, |v_2\rangle, \ldots$ に対応する．

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，$|v_j\rangle$ 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，入力ベクトルは $|v_2\rangle, |v_3\rangle$ の 2 つである．したがって，式 (5) のように線形結合で $|t\rangle$ を表現できる．

$$|t\rangle = \frac{1}{2+\omega}|v_2\rangle + \frac{1+\omega}{2+\omega}|v_3\rangle \tag{5}$$

2.2.2 SPQA の概要

SPQA は，Span Program をベース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 とŠpalek 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おり [8]，ここで提案されているグラフ作成の方法を述べる．

任意の論理式を計算する Span Program P について，ターゲットベクトルを $|t\rangle$，入力ベクトル $|v_i\rangle \in V$ を各列の要素として持つ行列を A，入力ラベルの集合を I，$[n] = \{1, 2, \ldots, n\}$，$\Pi(x)$ とし，それぞれ以下のように表す．

$$A = \sum_{i \in I} |v_i\rangle\langle i| \in \mathcal{L}\left(\mathbb{C}^{|I|}, V\right) \tag{6}$$

$$I = \bigcup_{j \in [n], b \in \{0,1\}} I_{j,b} \tag{7}$$

$$\Pi(x) = \sum_{i \in I(x)} |i\rangle\langle i| \in \mathcal{L}\left(\mathbb{C}^{|I|}\right) \tag{8}$$

$$I(x) = \bigcup_{j \in [n]} I_{j,x_j} \tag{9}$$

A と V_J はおおよそ同義であるが，V_J は A の不必要な部分を除いた行列である．$\Pi(x)$ は一般的に射影行列と呼ばれる．ここでは，射影は行っていないが，任意の行列に右から作用することにより，その行列の $i \in I(x)$ 番目の列を残し，他の列を 0 にする．このときの重み付き完全 2 部グラフの隣接行列 $B_{G_P(x)}$ を次式のように表す [15]．

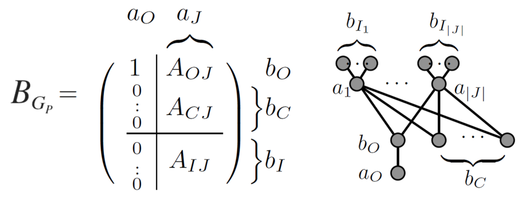

図 1 隣接行列 $B_{G_P(x)}$ によるグラフ構造の作成方法

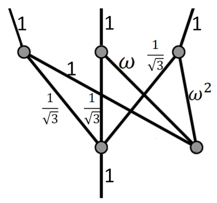

図 2 Maj_3 におけるグラフ構造

$$B_{G_P(x)} = \begin{pmatrix} |t\rangle & A \\ 0 & 1 - \Pi(x) \end{pmatrix} \tag{10}$$

$B_{G_P(x)}$ とグラフ構造の対応を図 1 に示す．A_{OJ} は A の 1 行目であり，A_{CJ} は A の 2 行目以降である．また，$A_{IJ} = 1 - \Pi(x)$ 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，ノード間の枝は無いものとみなす．例として Maj_3 の Span Program を式 (4) としたときの $B_{G_P(111)}$ の隣接行列を式 (11) に，グラフを図 2 に示す．

$$\left(\begin{array}{c|ccc} 1 & \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} \\ 0 & 1 & \omega & \omega^2 \\ \hline 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{array}\right) \tag{11}$$

2.3 量子クエリ計算量

量子アルゴリズムでは問い合わ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，問い合わ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，少ない問い合わ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]．このため，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．

$$Q(f) = O(wsize(P)) \tag{12}$$

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

$$wsize(P) = \max_{x \in \{0,1\}^n} wsize(P, x) \tag{13}$$

$f_P(x) = 1$ 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 $|w\rangle \in \mathbb{C}^{|I|}$ が存在し，$wsize(P,x)$ を式 (14) のように表すことができる．ここで，$\mathbb{C}^{|I|}$ は $|I|$ 次元の複素空間を表す．

$$wsize(P, x) = \min_{|w\rangle : A\Pi(x)|w\rangle = |t\rangle} || \, |w\rangle \, ||^2 \tag{14}$$

$f_P(x) = 0$ 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 $|w'\rangle \in V$ が存在し，$wsize(P,x)$ は以下のように表すことができる．

$$wsize(P, x) = \min_{\substack{|w'\rangle : \langle t|w'\rangle = 1 \\ \Pi(x) A^\dagger |w'\rangle = 0}} ||A^\dagger |w'\rangle \, ||^2 \tag{15}$$

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．

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 を導出する手法を提案する．定式化においては Reichardt らの導出した定理に基づいて行う [9]．Reichardt らは，一般量子対向界を導出する式を変換し，次式を導出している．この式の行列 X_j の集合 $\{X_1, X_2, \ldots, X_n\}$ を用いて，witness size が一般量子対向界に等しくなる Span Program を導出することができる．

$$Adv^\pm(f) = \min \max_{x \in B^n} \sum_{j \in [n]} \langle x| X_j |x\rangle \tag{16}$$

$$s.t. \quad \sum_{j : x_j \neq y_j} \langle x| X_j |y\rangle = 1 \ if \ f(x) \neq f(y) \tag{17}$$

$$X_j \succeq 0 \tag{18}$$

本論文で提案する方式では，式 (17) のもとで目的関数 F を式 (19) 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 (17) とする．

$$F = \begin{cases} \max_{x \in B^n} \sum_{j \in [n]} \langle x| X_j |x\rangle - Adv^\pm(f) \ if \ X_j \succeq 0 \\ P(X_j) \ otherwize \end{cases} \tag{19}$$

ここで，x，y は論理式の入力値を表し，$P(X_j)$ 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 (17) を，それぞれ式 (20)，式 (21) 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 (21) は，$k \leq l$ における制約条件を表す．

$$F = \max_{k \in \{00,01,10,11\}} \sum_{j \in \{1,2\}} X_{j,(k,k)} - \sqrt{2} \tag{20}$$

$$s.t. \quad \begin{cases} X_{1,(00,10)} = 1 \\ X_{2,(00,01)} = 1 \\ X_{1,(00,11)} + X_{2,(00,11)} = 1 \end{cases} \tag{21}$$

3.2 設計変数の削減

本方式では，$\{X_1, X_2, \ldots, X_j, \ldots\}$ の全ての要素を設計変数とするのではなく，式 (17) と，式 (18) 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．すなわち，制約が式 (17)（条件 A）および式 (18) を満たすことであり，後者が，$\{X_1, X_2, \ldots, X_n\}$ が対称行列（条件 B）かつ固有値が全て 0 以上であること（条件 C）に着目して，条件 A および B を満たすように設計変数を決定する．

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 (21)）より，式 (21) に示すように一部の成分が算出される．このため，式 (21) 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 は常に満たされるため，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

*1 式 (3) の $Span$ と $Range$ はおおよそ同義であるが，$A\Pi(x)$ のように行列であるか $|v_j\rangle$ のように列ベクトルであるかにより表現方法が異なる．

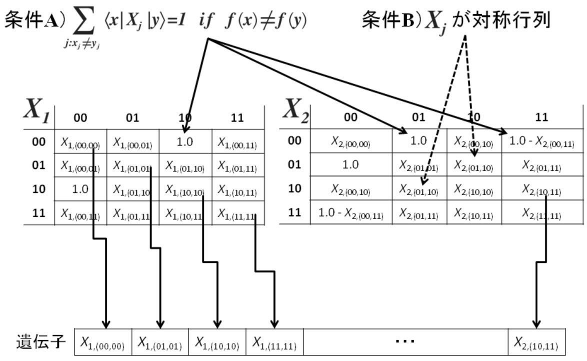

図 3 OR_2 における遺伝子表現

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．

$$P(X_j) = m \times T_{Pen} \tag{22}$$

4. 評価実験

4.1 概要

提案する方式により，最適な Span Program の導出を試みる実験を行った．4.2.1 節および 4.2.2 節では，それぞれ 2bit，3bit の論理式を対象として，DE を用いて実験を行った．なお，以下では $T_{Pen} = 1{,}000$ とした．

本実験では，発見した最良個体を Span Program に変換し，最適な Span Program であるかを評価する．すなわち，$\{X_1, X_2, \ldots, X_n\}$ を式 (23) のようにコレスキー分解することで $\{|v_{xj}\rangle\}$ を求める．

$$\{|v_{xj}\rangle\} : \langle v_{xj}|v_{yj}\rangle = \langle x|\, X_j \,|y\rangle \tag{23}$$

求めた $\{|v_{xj}\rangle\}$ を用いて式 (24) より行列 A を導出する [9]．

$$A := \sum_{x \in F_0, j \in [n]} |x\rangle \langle j, \overline{x_j}| \otimes \langle v_{xj}| \tag{24}$$

ここで，$F_0 = \{x : f(x) = 0\}$ である．

この行列 A より，式 (13)，式 (14)，式 (15) を用いて witness size を求める．この witness size と一般量子対向界，既知の最適解の witness size とを比較することで，最適な Span Program であるかを評価する．

4.2 実験結果

4.2.1 2bit の論理式

2bit の論理式 OR_2，AND_2，XOR_2 を対象として，DE/rand/1/bin により Span Program の導出を試みた．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，交叉率 CR を 0.9，スケール係数 F_{SC} を 0.5 とした．試行回数を 5 回とした．

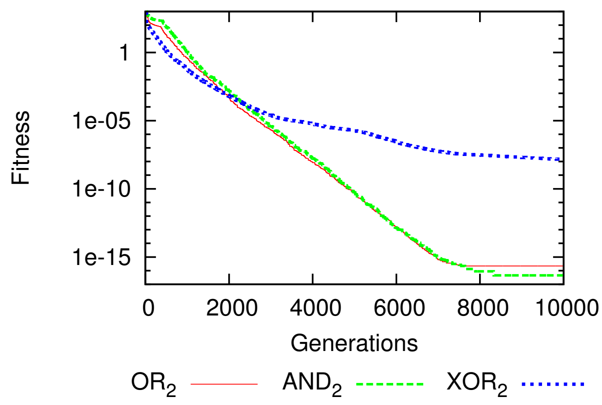

図 4 OR_2，AND_2，XOR_2 における適応度の推移グラフ

表 3 実験を行った 2bit の論理式における $Adv^{\pm}(f)$ と witness size

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（3bit）

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
$If\text{-}Then\text{-}Else$	$(x_2 \wedge x_3) \vee (x_1 \wedge \overline{x_3})$	92
$Equal_3$	$(x_1 \wedge x_2 \wedge x_3) \vee (\overline{x_1} \wedge \overline{x_2} \wedge \overline{x_3})$	96

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 より，いずれの論理関数においても世代数が大きくなるにつれて適応度が 0 に収束したことがわかる．

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 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わかる．

4.2.2 3bit の論理式

表 4 に示す 3bit の論理式を対象として，Span Program の導出を試みた．ここでは DE/best/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，交叉率 $CR = 0.9$，スケール係数 $F_{SC} = 0.3$ 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

表 5 実験を行った 3bit の論理式における $Adv^{\pm}(f)$ と witness size

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
$If\text{-}Then\text{-}Else$	2	2	2.00921
$Equal_3$	$3/\sqrt{2}$	$3/\sqrt{2}$	2.16242

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

まで $Adv^{\pm}(f)$ と等しい witness size の Span Program を導出できたことがわかる．

5. おわりに

SPQA で用いる 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 DE 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，一般的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．

今後は，最適解が未知の論理式に対する Span Program の導出や，多 bit の論理式に対する Span Program の導出を検討する．

参考文献

[1] Farhi, E., Goldstone, J. and Gutmann, S.: A quantum algorithm for the Hamiltonian NAND tree, *arXiv preprint quant-ph/0702144*, pp. 1–16 (2007).

[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, *Foundations of Computer Science, 1986., 27th Annual Symposium on*, IEEE, pp. 29–38 (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 1/2+ o(1)), *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 cur, 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 cur, R.: Span-program-based quantum algorithm for evaluating formulas, *Proceedings of the 40th annual ACM symposium on Theory of computing*, ACM, pp. 103–112 (2008).

[9] Reichardt, B. W.: Span programs and quantum query complexity: The general adversary bound is nearly tight for every boolean function, *Foundations of Computer Science, 2009. FOCS'09. 50th Annual IEEE Symposium on*, IEEE, pp. 544–551 (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] Reishardt, B. and Špalek, R.: Quantum query complexity of up to four-bit functions, http://www.ucw.cz/ robert/papers/gargets.

[12] Karchmer, M. and Wigderson, A.: On span programs, *Structure in Complexity Theory Conference, 1993., Proceedings of the Eighth Annual*, IEEE, pp. 102–111 (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.: 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, *Journal of the ACM (JACM)*, Vol. 48, No. 4, pp. 778–797 (2001).

[17] Szegedy, M., Barnum, H. and Saks, M.: Quantum decision trees and semidefinite programming (2001).

[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 cur, R.: Negative weights make adversaries stronger, *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, ACM, pp. 526–535 (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).