数理解析研究所講究録 1221 巻 2001 年 70-89

ソリトンセルオートマトンと量子コンピューティング

由良文孝 (東大数理) *

概 要

ソリトン方程式の超離散極限として得られる可積分な箱玉系を、多項式サイズの量子回路として実現した。その過程で、周期的境界条件を課した有限箱玉系を考察することにより、論理漸化式の表現を初めて得た。さらに、この漸化式が $2N$ 乗根を求める可積分アルゴリズムに対応することを示した。

1 はじめに

近年、量子コンピュータが応用上重要であることが認識されて注目を集めている。1994年 Shor によって因数分解が多項式時間で解けることが示されて以来のことである [1]。

計算理論は 1930 年代に始まる。Gödel の不完全性定理に始まり、Church が計算可能関数とは帰納的関数のことであると提唱、Turing が Turing 機械と呼ばれるコンピュータの数学モデルを定義した。その後 40 年経ち、70 年代にコンピュータが量子力学の法則に従い可逆ならば、どのようなものになるだろうか?という問いがなされた。可逆計算は、計算素子の熱力学的効率＝計算過程の物理への興味として始まった [2, 3]。本質的に量子力学系は可逆な物理系であり、理想的なモデルであることから量子計算の研究がなされた。量子コンピュータの数学的モデルは、1985 年に Deutsch の考えた量子 Turing 機械 (QTM) として一般化されている [4]。

多くの自然現象は、時間空間変数、物理量が連続である。現象の理解のためにシミュレーションが行われることが多いが、そのために離散的なモデルの構築が有用な場合がある。用いられる離散的なモデルは、離散化、超離散化の手法のなかで連続的なモデルの数理構造を保つことが必要であるが、可積分系はその代表例である。ソリトン方程式のなかでもっとも基本的な KP 方程式から出発し、離散的なソリトンセルオートマトンまで導くことができる。また逆に可積分系は良い (古典的な) アルゴリズムに対応している [6, 7]。例を挙げると、行列の対角化 (QR アルゴリズムなど)、数列の加速法 (ε アルゴリズムなど)、暗号の復号化 (BCH-Goppa 符号の復号化)、線形計画法 (Karmarkar の内点アルゴリズム) など様々なアルゴリズムが非線形可積分系の発展方程式と等価であることが知られている。「可積分アルゴリズム」は量子コンピュータにおいて、どのような役割を果たすであろうか? これまでに知られている量子アルゴリズムは未だ数少ない。代表的なものとして、Shor の因数分解アルゴリズム [1] と Grover のデータベース検索が挙げられる。

*E-mail : yura@poisson.ms.u-tokyo.ac.jp

本研究では新しい量子アルゴリズムの発見や、具体的な問題への応用を目指して箱と玉の系の論理式を導き、その量子回路を具体的に構成した。2章では、離散可積分系として周期的な箱玉系 (periodic Box and Ball System, pBBS) を取り上げる。可積分系を実装するにあたり有限な系が簡単であろうと思われるからである。ただし周期的境界条件をとり有限系にすることは量子コンピュータにとって本質的ではない。量子コンピュータはQTMと同等であるため、TMの拡張として有限制御部と可算無限長のテープの上で定義されているからである [5]。しかしながら周期的箱玉系は、それ自体興味深い対象であり、その構成法と初期状態の分類について得られた結果にふれる。3章では、ブール代数の上に箱玉系を構築し、箱玉の漸化式を与える。さらに、この論理式の逆超離散が $2N$ 乗根を求めるアルゴリズムに対応していることを示す。この系は保存量を持ち、その振る舞いから見て (古典) 可積分アルゴリズムになっている。4章では量子計算の導入として、一般的な枠組みを古典計算と比べながら述べる。最後に5章で、箱玉系の量子回路の実装を行い、その計算量を見積もる。その結果、多項式時間 $O(N^2)$ での箱玉系の量子回路を得た。

2 離散可積分系

超離散化の操作は、方程式の従属変数を離散化する。つまりセルオートマトン (Cellular Automaton, CA) を与える。ここではソリトン方程式から得られる「箱と玉の系 (Box and Ball System, BBS)」の概略を述べ、周期的なBBS(pBBS) を定義する。

2.1 無限箱玉系

最初に、1990年高橋と薩摩によって提案された箱と玉の系を解説する [8, 9]。まず同じ玉を N 個用意し、その玉を入れる箱を無限個用意して一列に並べる。ただし1つの箱に玉は1つしか入らないものとする (ここでは考慮しないが、この制限を拡張したnonautonomousなモデルとして [10] などがある)。この箱の列にすべての玉を適当に入れて初期状態とする。系の時間発展ルールを次のように与える。時刻 t から時刻 $t+1$ への時間発展は、

「すべての玉を1回ずつ、左から順に、右方の空き箱へ移す」

とする。この時間発展例を図1に示す。時間を経てもソリトン的なふるまいを保つことが

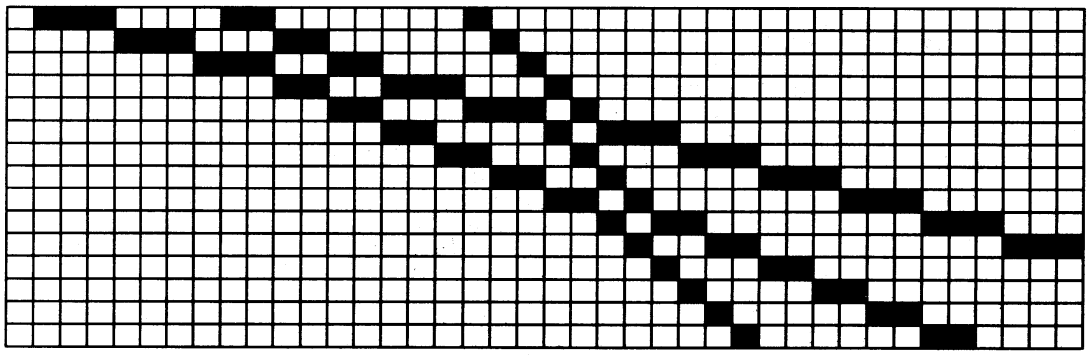

図 1: BBS の時間発展 (時間の向きは上から下)

見て取れる。また、無限個の保存量を持つことが知られている [11]。

このBBSは、「運搬車」というパラメータを導入することにより、一般化できる [12]。運搬車は最初玉を積んでいないとし、左から右へ走り抜ける。この運搬車の最大積載量を玉 M 個とおく。この時、無条件に玉を右方の空き箱へ移していたBBSルールを箱と運搬車の相互作用を用いて、以下のように変更する。運搬車が箱を通過する際に、

1. 箱に玉が入っているならば
   (a) 運搬車に空きがあれば、玉を載せる。
   (b) 運搬車に空きがなければ、そのまま通過する。
2. 箱に玉が入っていなければ
   (a) 運搬車に玉があれば、玉を降ろす。
   (b) 運搬車に玉がなければ、そのまま通過する。

この運搬車付きBBSの時間発展例 ($M = 3$) を図2に示す。この定義からわかるように $M = \infty$ のBBSが、運搬車のないBBSに対応する。また $M = 1$ の時は、玉のパターンは箱1つ分、単に右シフトする。つまり、M はソリトンの速度の上限を与えていることがわかる。

BBSは可解格子模型で表現できることが知られている [13]。$U_q(\widehat{sl_2})$ の組み合わせ論的 R 行列で構成される可解格子模型を考えると、箱の状態が B_1、運搬車 (積載量 M) が B_M に対応し (B_l は l 次対称積表現)、基底状態のパターンがBBSの時間発展を与える (図3)。ここで箱に玉が入った状態を"1"、空の状態を"0"で表わしている。箱の中身の時間発展は、$B_1^{\otimes\infty} \to B_1^{\otimes\infty}$ で与えられる。この時、運搬車の初期状態および終状態では玉は積んでいない (最高ウェイトベクトル)。これは無限BBSに、十分遠方の箱では空であるという境界条件を課していることと対応している。

2.2 周期的BBS(periodic BBS, pBBS)

前節のBBSは、無限遠の彼方から来たソリトンが互いに相互作用して再び無限遠に去っていく過程を表わしていた。しかし周期的境界条件のもとでは、1度去ったソリトンが再

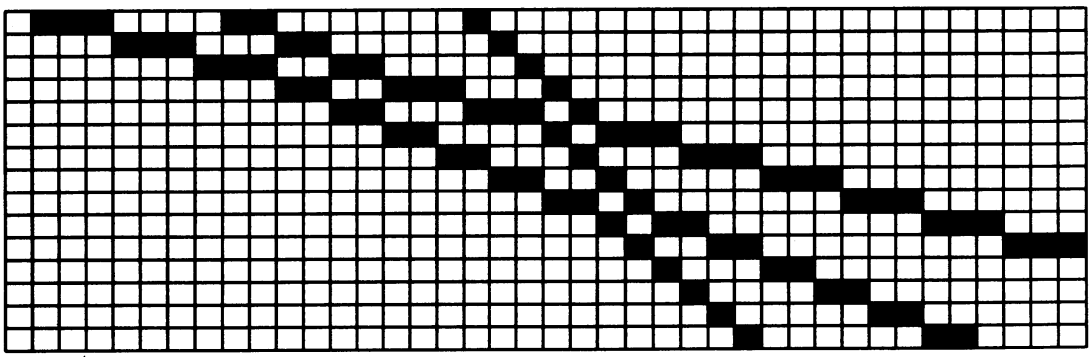

図 2: 運搬車付きBBS(積載量 $M = 3$)

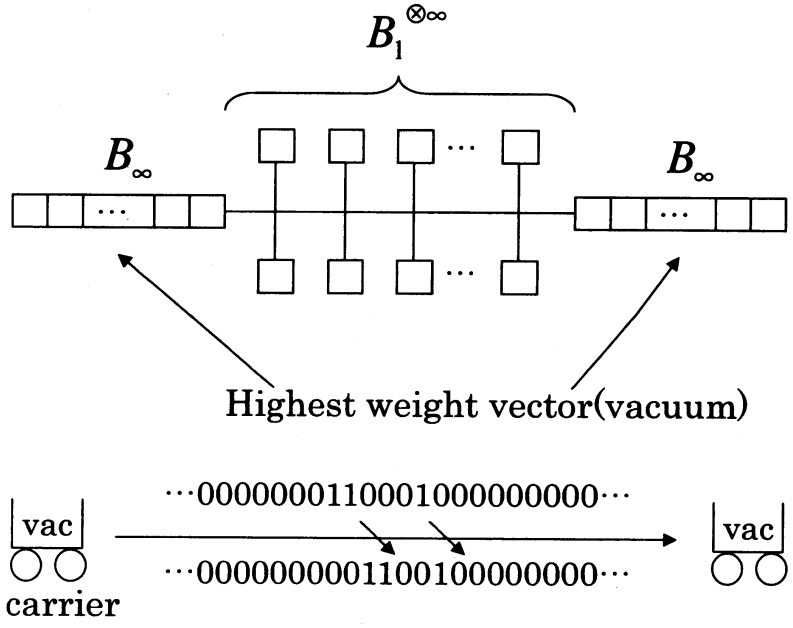

図 3: 可解格子模型と、対応する無限 BBS

び戻り相互作用する。本節では、BBS を周期系に拡張する。

前節の無限系の場合と違い pBBS は端を周期的境界条件でつないでいるため、運搬車の初期状態が「空」では、一般に正しい時間発展を与えない。図 4 に、pBBS のパターン"00001"(箱の数 $N = 5$) に運搬車を作用させた場合の例を示した。この例の場合、運搬車の初期状態として"1"であった場合のみが正しく BBS を再現している。一般に pBBS(箱の数 N) では、$M : B_\infty \otimes B_1^{\otimes N} \to B_1^{\otimes N} \otimes B_\infty$ の同型写像から、運搬車の link にトレースを取って $\mathrm{Tr}_{B_\infty} M : B_1^{\otimes N} \to B_1^{\otimes N}$ とすれば pBBS の時間発展が得られる [14]。

ここで玉の数は箱の数の半分以下とする。玉の数が空き箱の数より多い時は、"1"↔"0"を入れ替えれば「粒子」↔「反粒子」のように対応がつく。この反粒子の動きは粒子とは逆で、右から左へと動く。このルールは無限 BBS にはなかったものである。

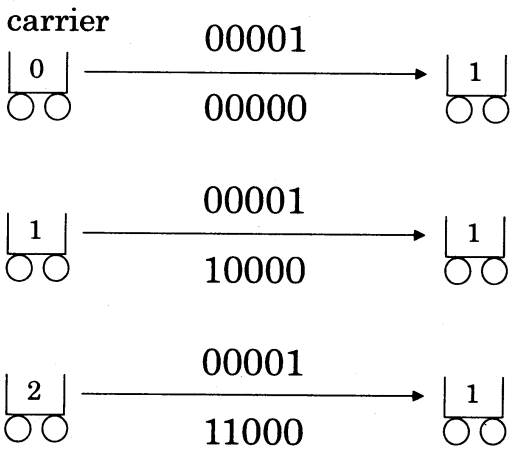

図 4: 運搬車の初期状態は終状態と等しくならなければいけない。

運搬車 (積載量 n) 付き pBBS(箱の数 N) による 1 単位時間の時間発展を置換写像 $T_n^{(N)}$ と定義する。無限箱玉系のときと同様、$T_\infty^{(N)}$ が運搬車なしの BBS、$T_1^{(N)}$ が 1 ビットの回転右シフト (rotate right shift) に対応する。肩の N は以降省略する。pBBS のルールより、$n \geq 1$ のとき T_n の固定点は、"$\overbrace{000\ldots0}^{N}$"と"$\overbrace{111\ldots1}^{N}$"のパターンのみである。

周期的箱玉系を得る別の方法として、箱の数を N としたとき、これを $N-1$ 種類の玉に対応させ、$\widehat{sl_N}$ から可解格子模型を作ることもできる (図 5)[14]。

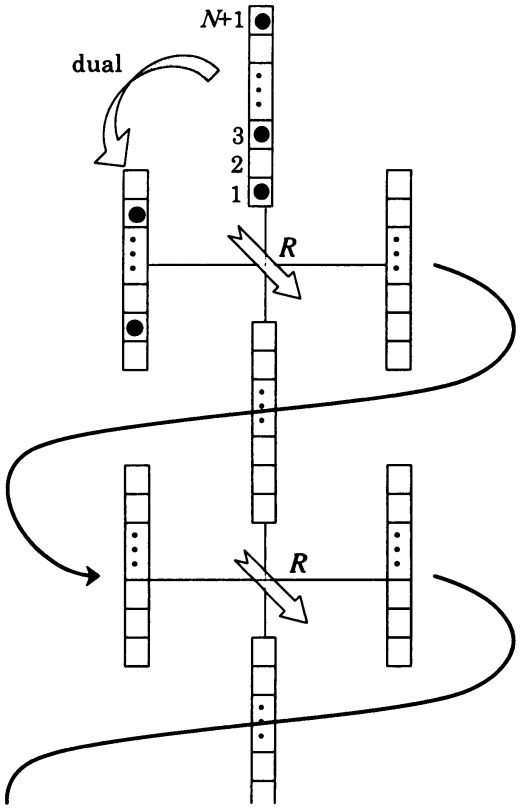

図 5: $\widehat{sl_N}$ による可解格子模型と BBS

2.3 周期的箱玉の例とその分類

pBBS(T_∞) の例を図 6 に示す (箱の数 $N=7$)。(a) は初期状態"1110000"から時間発展させたものであり、長さ 3 のソリトンが、速さ 3 で進んでいる様子がわかる。(a) は明らかに周期 7 を持つ。これに対して (b) の初期状態"1101000"からの時間発展は、3 単位時間後に 1 サイト右シフトしたパターンが表れている。つまりこの"11"と"1"の 2 ソリトンからなる初期状態は周期 21 を持つ。

一般の N でのこの周期の規則はよくわかっていない。$N=9$ の表をその例として挙げる (表 1)。"1"の数が"0"の数よりも多くないときのみを考えればよいので、総計 $2^{N-1}=256$ パターンの分類となっている表の列は左からそれぞれ、初期状態、玉の数、ソリトンの数、この初期状態に属する周期となっている。またルールから明らかに、玉とソリトンの数は保存している。

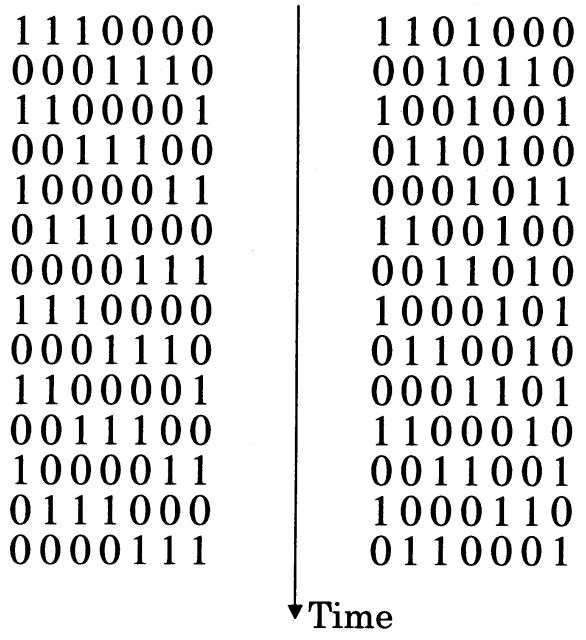

図 6: $N = 7$ の pBBS の例。 初期状態は (a) "1110000" (b) "1101000"

BBS の初期状態は、回転右シフト T_1 のもとで分類できる。なぜならば、T_∞ は $T_1 T_\infty = T_\infty T_1$ をみたし、シフトに対して可換だからである。一般に $T_m T_n = T_n T_m (m, n \in \mathbf{Z})$ を満たすことを可解格子模型の立場から示すことができる。可解格子模型の背景には Yang-Baxter 方程式を満たす R 行列があり、異なるパラメータに属する転送行列が交換するからである [15]。

さて例えば、箱の数 $N = 6$ のとき初期状態"100100"から時間発展させることを考える。明らかに、この"100100"のパターンの時間発展は、$N = 3$ の初期状態"100"のパターンの時間発展を含んでいる。一般に箱数 N に含まれるあるパターンは、N の約数の箱数のあるパターンを含んでいることがわかる。この整除関係による包含関係は整数 N の Hasse 図に同型である。つまり、$N = 2^{e_1} 3^{e_2} 5^{e_3} \cdots$ と素因数分解したときの $(e_1, e_2, e_3, \ldots)$ の整除関係で成す半順序関係で分類できる。この関係を利用すると例えば、N シフトして初めてもとに戻る $(T_1^N x = x, x \in \{0,1\}^N)$ パターンが何通りあるかがわかる。そのパターン数は、Möbius 関数 $\mu(x, y)$ とその反転公式を用いて、

$$\sum_{x<N} 2^x \mu(x, N)$$

と表わすことができる。ここで記号 $<$ は、N の整除関係から得られる半順序集合の上の順序である。例えば $N = 6$ のとき 6 シフトして初めて元に戻るパターン数は、$2^6 - 2^3 - 2^2 + 2^1 = 54$(通り) である。

また詳しくは述べないが、巡回群 $C_N = \left\{e, T_1, T_1^2, \ldots, T_1^{N-1}\right\}$ の巡回置換指数を用いることにより、シフトして一致するパターンのものを数え上げることもできる (Pólya の定理 [17])。

T_∞ が $T_1 T_\infty = T_\infty T_1$ を満たすことを用いて、シフトして重なる初期状態について分類

Initial State	Ball	Soliton	Period
000000000	0	0	1
100000000	1	1	9
110000000	2	1	9
101000000	2	2	9
100100000	2	2	9
100010000	2	2	9
111000000	3	1	3
011100000	3	1	3
001110000	3	1	3
101100000	3	2	15
110100000	3	2	15
110010000	3	2	15
101010000	3	3	9
101001000	3	3	9
100101000	3	3	9
100100100	3	3	3
111100000	4	1	9
110011000	4	2	9
111010000	4	2	45
110101000	4	3	27
101101000	4	3	27
101010100	4	4	9
計			2^8=256

表 1: $N = 9$ の pBBS(T_∞) の初期状態の分類

することができた。pBBS は、本質的に組み合わせ論的考察を必要とする。現段階では先に述べたように、一般的な箱数 N の場合の $T_\infty^{(N)}$ の巡回置換の長さ (=箱玉系の周期) についてはよくわかっていない。

3 箱玉論理式とアルゴリズム

3.1 箱玉論理式

前節で導入した pBBS が論理式で書けることを示す。$\wedge$、$\vee$、$\oplus$ をそれぞれ AND、OR、ExOR(排他的論理和) とする。このとき演算はビット列に対しても同様に、ビットごとの演算を行うものとする ($X = (x_1, x_2, \ldots, x_N)$, $Y = (y_1, y_2, \ldots, y_N) \in \mathbf{Z_2}^{\mathbf{N}}$ として、$(X \wedge Y)_i = x_i \wedge y_i$ など)。この N 桁ビット列を pBBS の箱の状態とみて、玉の入っている状態を真 ("1")、入っていない状態を偽 ("0") として、pBBS とビット列を対応付ける。そうすると、積載量 1 のときの pBBS(T_1) は

$$T_1 X = (x_N, x_1, x_2 \ldots, x_{N-1})$$

と作用する (rotate shift)。さらに、**積載量** ∞ のときの pBBS(T_∞) は次のようになる。

定理 (pBBS の論理式表現)
$X(\in \mathbf{Z_2}^{\mathbf{N}})$ が与えられたとき、$T_\infty X$ を求めるものとする。$A_0 = B_0 = X$ として漸化式

$$\begin{cases} A_{n+1} = A_n \vee B_n \\ B_{n+1} = T_1(A_n \wedge B_n) \end{cases} \tag{1}$$

が $T_\infty X = A_N \oplus X, B_N = 0$ を与える。

BBS のルールとは"1"があれば、対応する"0"を右方に探すことであった。この漸化式の n 段目は、"1"から数えて n 個分離れた箱の中に"0"を探すことと対応している。漸化式の途中で、A_n, B_n はそれぞれ、

$$\begin{cases} A_n = X \oplus (\text{"0"から"1"へ変化したビット}) \\ B_n = X \oplus (\text{"1"から"0"へ変化したビット}) \end{cases}$$

を表わしている。例えば式 (1) に X = "1101000" を与えると A_7 = "1111110", B_7 = "0000000" となり、$T_\infty X$ = "0010110" を得る。ビット列 B_n の中で、すべての"1"が"0"に変化すれば玉の移動は終わる。

この漸化式は AND, OR, SHIFT の 3 演算のみで表わされ、非常に単純で基本的な形をしている。各論理式に AND, OR が 1 回ずつ現われ、ソリトンの左右対称性の破れ (ソリトンの左から右への動き) を導入するための最低限のシフト演算 T_1 が 1 回入っているだけだからである。

系 (pBBS の論理式表現 (2 倍の高速化))
$X(\in \mathbf{Z_2}^{\mathbf{N}})$ が与えられたとき、$T_\infty X$ を求めるものとする。$A_0 = X, B_0 = T_1 X$ として漸化式

$$\begin{cases} A_{n+1} = A_n \vee B_n \\ B_{n+1} = {T_1}^2(A_n \wedge B_n) \end{cases} \tag{2}$$

が $T_\infty X = A_{\lfloor N/2 \rfloor} \oplus X, B_{\lfloor N/2 \rfloor} = 0$ を与える。

式 (1) に従って玉を動かしていくと、"1"と動いた先"0"のペアの組み方が必ず入れ子構造になる。そこで式 (1) において、奇数の n のとき必ず $A_{n+1} = A_n, B_{n+1} = B_n$ であることから、式 (2) が導かれる。

3.2 可積分アルゴリズムとしての箱玉論理式

前節で、pBBS の論理式を得た。この式 (1) は簡単な形をしている。このソリトンを作る論理式は、いわゆる可積分アルゴリズム [6, 7] となにか関係しているのだろうか。セルオートマトンの世界から実数の世界へ戻ることにする。

3.2.1 連続量の方程式

論理変数"0"と"1"を整数の0と1と見なすと、以下のような対応がつく。

$$\begin{cases} x \wedge y & \Longleftrightarrow & \min(x,y) \\ x \vee y & \Longleftrightarrow & \max(x,y) \end{cases}$$

つまり、2値をとるAND, ORをそれぞれmin, maxとみなすことができる。ここでmax, minはビットごとに取るもの(bitwise)とする。すると、式(1)は

$$\begin{cases} A_{n+1} = \max(A_n, B_n) \\ B_{n+1} = T_1 \min(A_n, B_n) \end{cases}$$

と、整数で閉じた式に書き換えられる。そこでこの従属変数A_n, B_nを逆超離散化すると実数の上の方程式に戻ることができる。(逆)超離散化とは次の極限操作のことである。

$$\max(x,y) = \lim_{\epsilon \to +0} \epsilon \log\left(e^{x/\epsilon} + e^{y/\epsilon}\right)$$

$\min(x,y) = -\max(-x,-y)$に注意して、次式を得る。

$$\begin{cases} a_i^{(n+1)} = \dfrac{a_i^{(n)} + b_i^{(n)}}{2} \\ b_i^{(n+1)} = 2\left\{\left(a_{i-1}^{(n)}\right)^{-1} + \left(b_{i-1}^{(n)}\right)^{-1}\right\}^{-1} \end{cases} \tag{3}$$

ここで、ビット列の空間座標$i \in \{1,2,\ldots,N\}$をあらわに書いて、$a_i^{(n)} = e^{(A_n)_i/\epsilon}$, $b_i^{(n)} = e^{(B_n)_i/\epsilon}$と置いた。定数2は逆超離散化の過程の自由度の中から、式(3)が発散しないように決めた。整理すると、

$$\begin{cases} a_i^{(n+1)} = \left\{a_i^{(n)} + b_i^{(n)}\right\}/2 \\ a_{i-1}^{(n+1)} b_i^{(n+1)} = a_{i-1}^{(n)} b_{i-1}^{(n)} \end{cases} \tag{4}$$

となる。この式(4)が論理漸化式(1)に対応する、従属変数が連続な方程式である。

3.2.2 既存のアルゴリズムとの類似

ところで式(3)は空間座標iを無視すれば、

$$\begin{cases} a^{(n+1)} = \dfrac{a^{(n)} + b^{(n)}}{2} \\ b^{(n+1)} = \dfrac{2a^{(n)}b^{(n)}}{a^{(n)} + b^{(n)}} \end{cases} \tag{5}$$

となるが、これは算術調和平均のアルゴリズムとして知られ、$\lim_{n\to\infty} a^{(n)} = \lim_{n\to\infty} b^{(n)} = \sqrt{a^{(0)}b^{(0)}}$と、初期値の幾何平均を与えるものである。

また、式(4)は、QD(Quotient difference)アルゴリズム[16]と呼ばれる次式と酷似している。

$$\begin{cases} q_i^{(n)} + e_i^{(n)} = q_i^{(n+1)} + e_{i-1}^{(n+1)} \\ q_i^{(n)} e_{i-1}^{(n)} = q_{i-1}^{(n+1)} e_{i-1}^{(n+1)} \end{cases} \tag{6}$$

適当な座標変換のもとで、第2式は互いに一致している。QDアルゴリズムは有理関数の極をTaylor展開の係数から計算するアルゴリズムで、戸田分子方程式に対応することが知られている。

3.2.3 箱玉アルゴリズム

式 (4) には初期値を $\left\{a_i^{(0)}, b_i^{(0)}\right\}$ として、n に対する保存量

$$C^{(n)} = C^{(n-1)} = \cdots = C^{(0)} = \prod_{i=1}^{N} a_i^{(0)} b_i^{(0)} \equiv C \tag{7}$$

が存在する。この保存量に着目すると、各々の変数が保存量 C の $2N$ 乗根に収束することがわかる。

$$\lim_{n\to\infty} a_k^{(n)} = \lim_{n\to\infty} b_k^{(n)} = \sqrt[2N]{\prod_{i=1}^{N} a_i^{(0)} b_i^{(0)}} = \sqrt[2N]{C} \quad (\text{すべての } k \text{ に対して})$$

つまり、pBBS の論理式は $2N$ 乗根を求める可積分アルゴリズムと関係することがわかる。本論文の最初に、超離散化の手法では離散的なモデルが連続的なモデルの数理構造を保つことが必要であると述べた。ここで保たれている保存量 C に対応する、論理漸化式 (1) における量を見るには、C を前節とは逆に超離散化すればよい。すると、

$$C = \prod_{i=1}^{N} a_i^{(0)} b_i^{(0)} \quad \underset{\Longrightarrow}{\text{UD}} \quad \sum_{i=1}^{N} \{(A_0)_i + (B_0)_i\} \tag{8}$$

となる。この右辺は N 桁ビット列 A_0, B_0 に含まれる"1"の数の和であるから、玉の総数の 2 倍である (初期値 A_0, B_0 は X であることに注意)。確かに pBBS では玉の総数は保存量である。pBBS の (箱に玉が含まれる割合)$\equiv \dfrac{(\text{玉の総数の 2 倍})}{2N}$ の分母 $2N$ が、連続の世界では $2N$ 乗根として現れる仕組みになっている。

4 量子計算のモデル

ここまで有限系である pBBS について述べてきた。この系を量子回路として表現するために、まず量子計算のモデルについてまとめる。

4.1 古典計算と量子計算

量子系で古典計算を模倣する際に注意すべき点は、量子力学の時間発展がユニタリ変換を用いて記述されることにある。(ここでは測定過程に伴う波束の収縮や、デコヒーレンスは考えないことにし、純粋状態のみをとりあげる。そこで以降、波動関数で系を記述する。) 例えば全系のエントロピーなどはユニタリ変換で不変である。つまり、古典的な回路での 2 入力 1 出力 AND のような可逆でない計算はできないことになる。

古典計算では通常 0 と 1 を電圧の On, Off に対応させ、1bit とする ($\mathbf{Z_2} \equiv \{0, 1\}$)。そこで、置換 $f : \mathbf{Z_2}^{\mathbf{N}} \to \mathbf{Z_2}^{\mathbf{N}}$ ($\mathbf{Z_2}^{\mathbf{N}} \equiv \left\{0, 1, \ldots, 2^N - 1\right\}$) を考え、この計算を量子力学の枠組みで考えてみる。この状態に対応する量子計算基底として 2 次元 Hilbert 空間の正規直交基底 $\mathcal{Z}_2 \equiv \{|0\rangle, |1\rangle\}$ を用意する。例えばスピン 1/2 の粒子などである。この量子版のビットは qubit(quantum bit) と呼ばれる。$\mathbf{Z_2}^{\mathbf{N}}$ に対しても同様に N 個の qubit を用いて状態を張ればよい。必要な基底は $\mathbf{Z_2}^{\mathbf{N}}$ の表現に Nbit 必要であったのと同

じく、Nqubit あればよい ($\mathcal{Z}_{\mathbf{2}}{}^{\mathbf{N}}$)。以後簡単のために $||n\rangle \equiv \otimes_{i=0}^{N-1} |a_i\rangle$ と表わすことにする ($n = \sum a_i 2^i \in \mathbf{Z_2}{}^{\mathbf{N}}, a_i \in \{0,1\}$)。

上のように対応させると、置換 $n \mapsto f(n)$ は量子状態を用いて、

$$||n\rangle \ \mapsto \ ||f(n)\rangle \tag{9}$$

と表わせる。置換の入力と出力が 1 対 1 に対応することから、この状態変化はユニタリ変換を用いて表わすことができる。つまり関数 f を模倣するユニタリ変換 U_f が存在することがわかる。

$$U_f \ : \ ||n\rangle \ \mapsto \ U_f||n\rangle \ = \ ||f(n)\rangle \tag{10}$$

初期状態 $||n\rangle$ を準備して、あらかじめ用意したユニタリ変換 U_f (f の量子プログラム) を作用させると、出力 $||f(n)\rangle$ を得る。この出力を測定すれば、古典計算 $n \mapsto f(n)$ が量子計算の結果として求まることになる。

置換でない一般の写像の場合 (上で挙げた 2 入力 1 出力 AND など) には以下のように拡張できる。入力と出力が 1 対 1 でない場合には、あらかじめ空間を広げておいてその部分空間のみを用いればよい。例えば基底の数を倍に取り、そこでのユニタリ変換を

$$V_f \ : \ ||n\rangle \otimes ||0\rangle \ \mapsto \ ||n\rangle \otimes ||f(n)\rangle \tag{11}$$

とすることによって、1 つ目の状態ベクトルで入出力を 1 対 1 に保存し、2 つ目の状態が演算結果を保持するわけである。

これらの構成から、古典計算が可能な関数は量子計算が可能である。また逆に、少し意外な結果として、量子計算可能な関数は古典計算が可能であることも示されている [4]。量子計算が古典計算と本質的に異なるのは、量子系が状態の重ねあわせを許す点である。前述の U_f は重ね合わせ状態に対して

$$\begin{aligned} U_f \left\{ \sum_{n=0}^{2^N-1} c_n ||n\rangle \right\} \ &= \ \sum_{n=0}^{2^N-1} c_n U_f ||n\rangle \\ &= \ \sum_{n=0}^{2^N-1} c_n ||f(n)\rangle \end{aligned} \tag{12}$$

のように振舞う。その結果、ある出力 $f(n)$ が入力の重ね合わせ振幅 c_n に応じて確率 $|c_n|^2$ で得られる。特徴的なのは、1 回のユニタリ変換が 2^N 個の状態をそれぞれ $||n\rangle \mapsto ||f(n)\rangle$ と変換していることである。式 (12) を求めるには、古典計算だと 2^N 回の f の演算が必要であるが、量子計算では U_f を 1 回作用させるだけでよい。この並列性が、古典計算において指数時間かかる計算を多項式時間で解くポイントである。しかしながらこのままでは測定に応じて確率 $|c_n|^2$ で出力 $f(n)$ が得られるだけであり、重ね合わせによるメリットを活かした出力を得るためには、さらに工夫が必要となる。例えば、Shor のアルゴリズムに対する Jozsa の群論的アプローチ [18] などがある。箱玉系で有用な量子アルゴリズムを構成するためには、周期的箱玉系の性質を調べるとともにこういったアプローチが必要となるが、これは今後の課題である。

5 量子回路の実現

古典計算機は古典 TM として理解できるが、実際上は論理ゲートで構成される論理回路で表現する。これに対応して量子計算機でも、量子論理ゲートで量子論理回路を表現する。量子 TM と量子回路の対応については [5] に詳しい。

5.1 量子回路の基本ゲート

図 7 に基本的な量子ゲートを示す。1 本の線は 1 ビットを表わし、ゲートの左側が入力、右側を出力とする (つまり時間の向きは左から右)。NOT は 1 ビットの否定であり、$0 \leftrightarrow 1$ とフリップさせる。また、制御 NOT(Controlled NOT, CN) は、制御ビット (Control bit, 図では x) が"1"のときのみ標的ビット (Target bit, 図では y) の否定をとる。この標的ビットの出力は入力間の排他的論理和に等しい。

さてこれらは NOT のかわりに 1 ビットのユニタリ変換とすることによって一般化できる (図 7(c))。

$$U = \begin{pmatrix} u_{00} & u_{01} \\ u_{10} & u_{11} \end{pmatrix}$$

すると NOT はユニタリ変換の特殊な例であり、$\{|0\rangle, |1\rangle\}$ の基底で (基底は辞書式順序 (lexicographic order) にとる)

$$U_{NOT} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad (|1\rangle, |0\rangle) = U_{NOT}(|0\rangle, |1\rangle)$$

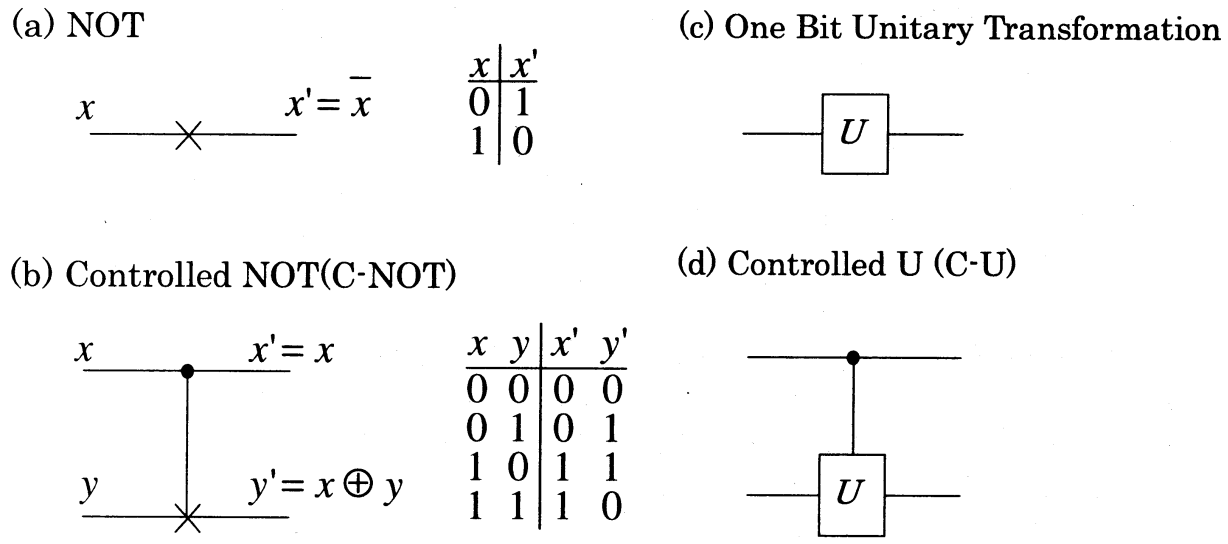

図 7: 基本的な量子ゲート

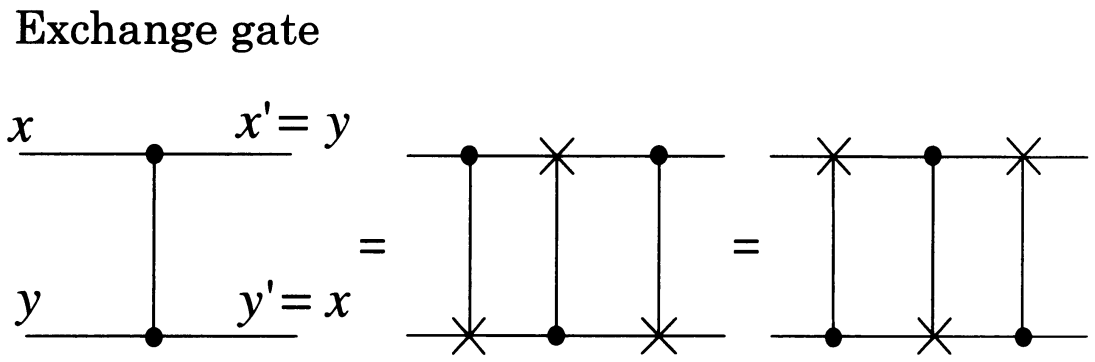

図 8: 量子交換ゲート

と表わせる (Pauli 行列 σ_x で書くことも多い)。同様に、制御 NOT についても

$$U_{CN} = \left(\begin{array}{cc|cc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \hline 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{array}\right)$$

$$(|00\rangle, |01\rangle, |11\rangle, |10\rangle) = U_{CN}(|00\rangle, |01\rangle, |10\rangle, |11\rangle)$$

となる。図 7(d) の制御 U については、1 ビット上の U を 3 個と制御 NOT を 2 個組み合わせて作ることができる [19]。さらに [19] は、この 1 ビット上の U と 2 ビット間の制御 NOT の組み合わせで任意の n ビット上のユニタリ変換を表わせることを示している。この二つのゲートを一般に「基本ゲート」と呼ぶ。

これらのゲートをさまざまに組み合わせ量子回路をつくる。例えば、図 8 のように U_{EX} : $(x, y) \mapsto (y, x)$ の交換ゲート

$$U_{EX} \equiv \left(\begin{array}{c|cc|c} 1 & 0 & 0 & 0 \\ \hline 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right) = \left(\begin{array}{cc|cc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \hline 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{array}\right) \left(\begin{array}{cc|cc} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ \hline 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{array}\right) \left(\begin{array}{cc|cc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \hline 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{array}\right)$$

や、図 9 の制御制御 NOT(Controlled-controlled NOT, C^2-NOT) なども作ることができる。ここで V は $V^2 = U_{NOT}$ をみたすユニタリ変換である。制御 ^{2}NOT は 2 つ制御ビットと 1 つの標的ビットからなり、制御ビットがすべて 1 の場合のみ、標的ビットに NOT を作用させるものである。そのため z を標的ビットとして、$U_{C^2-NOT} : (x, y, z) \mapsto (x, y, z \oplus (x \wedge y))$ と表わせる。後に示すように、この制御 ^{2}NOT は AND を作ることができる。

C²-NOT(controlled-controlled NOT)

x	y	z	x'	y'	z'
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	0	0
1	0	1	1	0	1
1	1	0	1	1	1
1	1	1	1	1	0

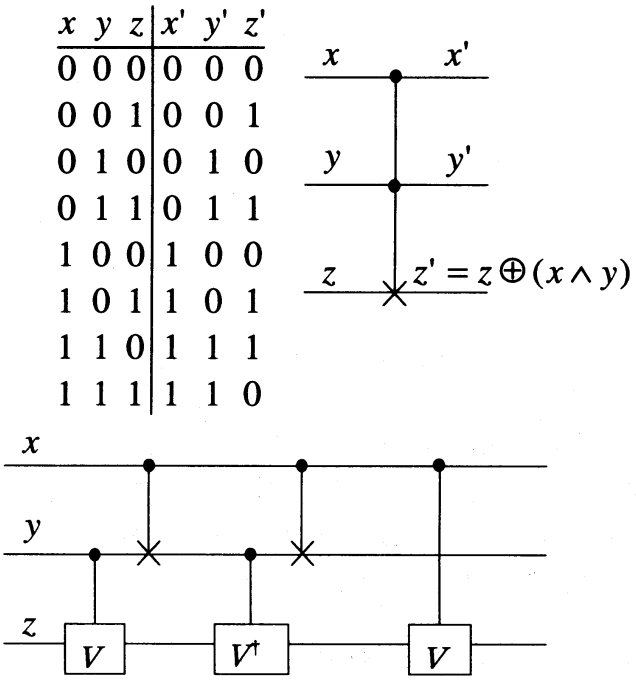

図 9: 制御制御 NOT(C^2-NOT)

5.2 置換としての pBBS

pBBS の時間発展は可逆である。そこで状態間遷移は置換として表わすことができる。例えば、$N = 3$ のときの具体例を挙げる。

$$\begin{cases} \text{"000"} \to \text{"000"} \\ \text{"100"} \to \text{"010"} \to \text{"001"} \to \text{"100"} \\ \text{"011"} \to \text{"110"} \to \text{"101"} \to \text{"011"} \\ \text{"111"} \to \text{"111"} \end{cases}$$

と時間発展していくので、"$a_2a_1a_0$" $\Leftrightarrow n \equiv \sum_i a_i 2^i$ と表わしてやると、

$$0 \to 0, 4 \to 2 \to 1 \to 4, 3 \to 6 \to 5 \to 3, 7 \to 7$$

に対応し、

$$\begin{pmatrix} 1 & 2 & 4 \\ 4 & 1 & 2 \end{pmatrix} \begin{pmatrix} 3 & 5 & 6 \\ 6 & 3 & 5 \end{pmatrix} \equiv (42)(21)(36)(65)$$

が箱の数 $N = 3$ のときの、pBBS の具体的な置換表現である。ここで演算順序は右から左にとっている。この置換式から制御 ^{2}-NOT を用いて、ただちに量子回路を作ることができる。図 10(a) は $8(= 2^3)$ 状態のうち、"011" $\leftrightarrow$ "111" のみを置き換えるから、この制御 ^{2}-NOT は置換 (37) であることがわかる。同様に (b) は (26)、(c) は (02) に対応することがわかる。ちなみに (c) の制御 ^{2}-NOT は、a_1 ビットが標的ビットである場合の表記である。一般に N ビットの回路における制御 $^{(N-1)}$NOT は、2^N 状態のうち 2 状態を互いに置

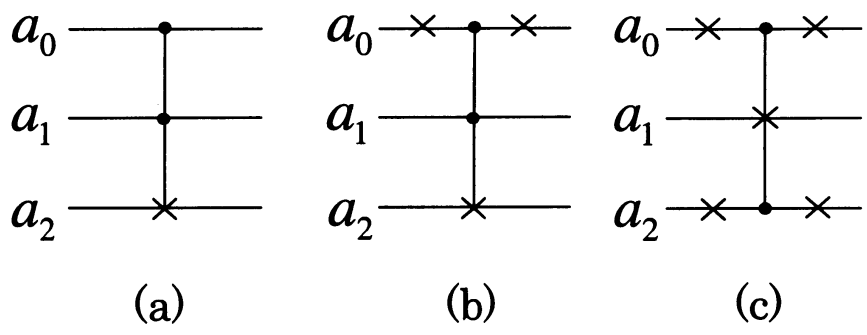

図 10: 制御 ^{2}NOT と置換

換し、それら 2 状態間の Hamming 距離は 1 である。ここで N ビット間で作ることのできる制御 $^{(N-1)}$-NOT の数は N 次元立方体の辺の数に一致することに注意する。これに対し BBS の状態遷移では、ある箱の中の玉が別の空箱の中に移るから、必ず Hamming 距離は偶数となる (これは玉の数が箱の数より多いときも成り立つ)。そこで

$$\begin{aligned}(42)(21)(36)(65) &= \underline{(02)(04)(02)}\ \underline{(02)(01)(02)} \\ &\quad \times \underline{(67)(37)(67)}\ \underline{(67)(57)(67)} \\ &= \underline{(02)(04)(01)(02)}\ \underline{(67)(37)(57)(67)}\end{aligned} \tag{13}$$

と変形すれば、対応する量子回路 (図 11) を得る。式 (13) をそのまま置きなおしたものが図 11(上) であり、NOT と制御 ^{2}NOT 間の交換関係で整理したのが図 11(下) である。この量子回路は、$N = 3$ のときの $|n\rangle \mapsto T_\infty |n\rangle$ を与える。量子回路であるからもちろん、入力状態の重ね合わせ $\sum_{n=0}^{2^N-1} c_n |n\rangle$ に対しては $\sum_{n=0}^{2^N-1} c_n |T_\infty n\rangle$ を与える。

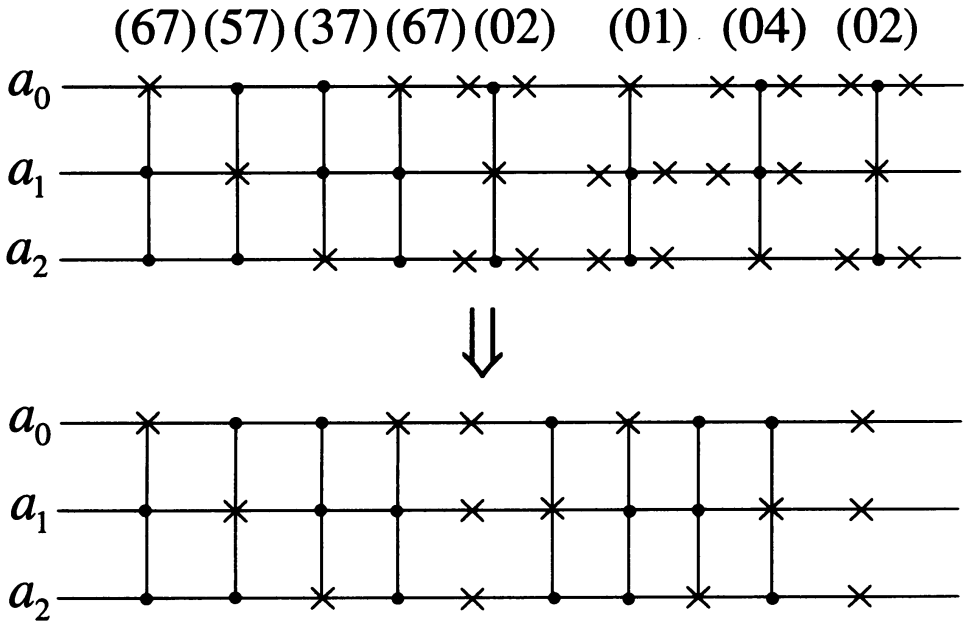

図 11: 置換による pBBS($N = 3$)

ここで回路長を考える。一般に量子回路の回路長とは基本ゲートの数と定義される。pBBS のルールで述べたように、置換 T_n の固定点は"000...0"と"111...1" のみであるから、この構成法により含まれる制御 $^{(N-1)}$NOT の数は $O(2^N)$ である。制御 $^{(N-1)}$NOT は $O(N)$ の基本ゲートで構成できるので [19]、結果として $O(N2^N)$ の回路長である。つまりこのアルゴリズムは、指数時間の計算量クラスであり、大きな N で実用にならない。また回路を作るために、2^N 個の状態にどう作用するかあらかじめ計算する必要がある。

ただし、N ビットの回路で必ずしも制御 $^{(N-1)}$NOT を使わずに、制御 $^{(m)}$NOT$(1 \leq m < N-1)$ を用いることもできる。その最適化は今後の課題であるが、一般に最適化問題は難しい (cf. [20])。ここで示した量子回路は、次節で述べる回路と (entanglement を除いて) 等価であるため、互いに何か関係づけられるかもしれない。

5.3 箱玉漸化式の量子回路

前節で示した回路は大きな N で実用にならない。そこで、箱玉漸化式 (1) を用いることを考える。そのために AND、OR、回転右シフト T_1 を量子回路で作る。AND は図 9 で入力 $z=0$ と制限すればただちに作れる。つまり 3 ビットの上で 4 状態に制限して可逆な AND を表現すればよい (cf. 式 (11))。同様に OR も 3 ビット上で表現できる (図 12)。

$$(x \wedge y) \oplus x \oplus y = x \vee y$$

であるから、$(x, y, 0) \mapsto (x, y, x \vee y)$ を得る。回転シフトについては、N ビット間で図 13 のように交換すればよい。以後、この T_1 の回路を右のように簡略化して示し、1 本の線で N ビットをまとめて書くことにする。その他量子回路についても、ビットごとに演算を行う。

これらをまとめると、漸化式の 1 ステップを作る量子回路を記述できる (図 14)。この回路をまとめて F と書く。$F : (A_n, B_n, 0) \mapsto (A_n, A_{n+1}, B_{n+1})$ は $3N$ ビットの入出力を持つユニタリ変換である。最終的にこの F を繰り返し適用することにより、漸化式を量子回路として表現できる (図 15)。ここで初期値 A_0, B_0 は、与えられた箱玉パターン X から制

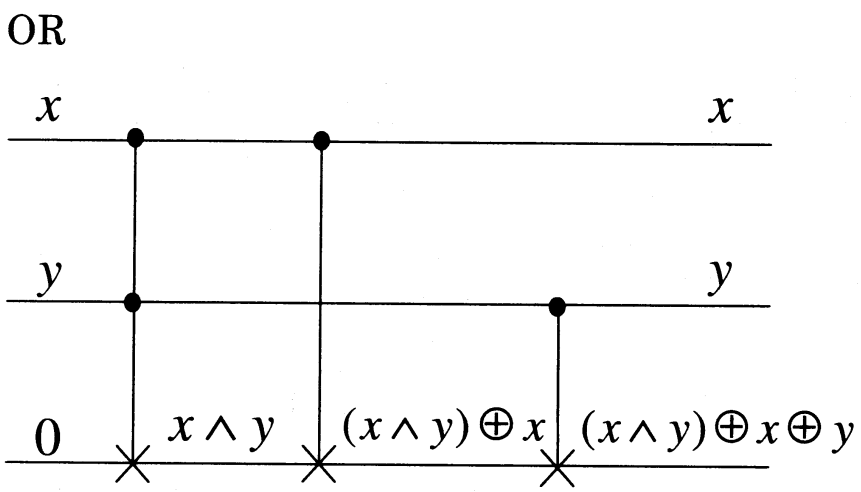

図 12: 量子 OR 回路

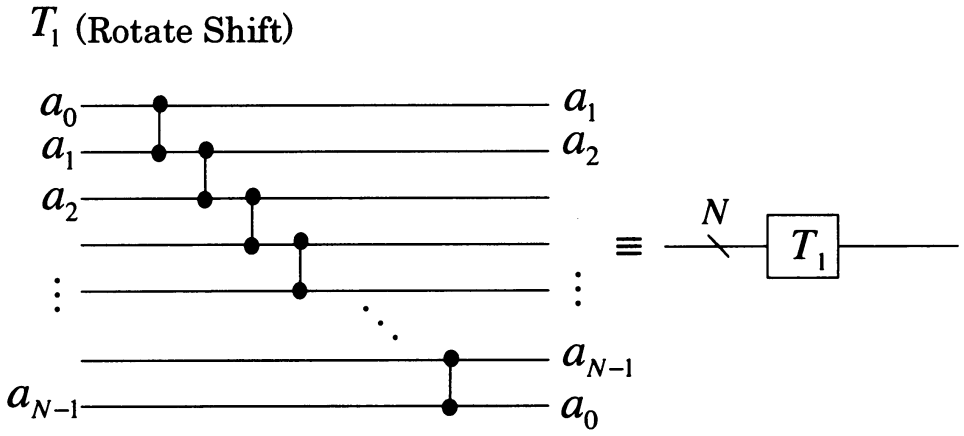

図 13: 量子回転シフト回路

御 NOT を用いて、

$$U_{NOT} : |X\rangle \otimes |0\rangle \mapsto |X\rangle \otimes |X\rangle$$

として得ている。

このままでは最終段で, A_N を得る過程で $A_1, A_2, \ldots, A_{N-1}$ の途中結果が残ってしまう。そのままでも何ら差し支えないが、図 16 のように工夫すると (図は $N = 2$ として 2 段まで描いてある)、途中にあらわれた結果を消去することができる。ここに F^{-1} は、回路 F を左右反転させて作られる回路で、ユニタリ行列 U_F の逆行列に対応している。量子力学

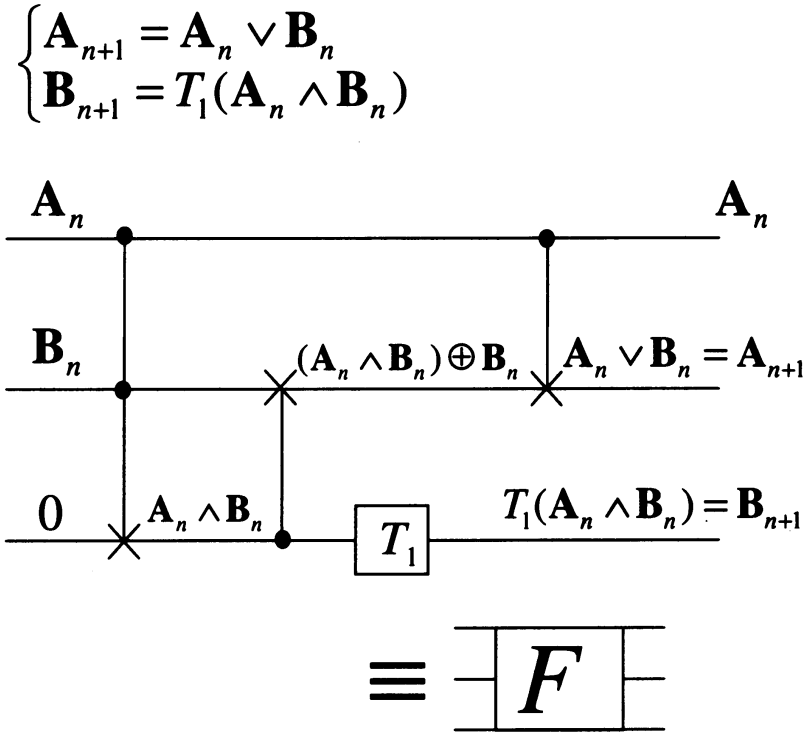

図 14: 漸化式の 1 段に相当する量子回路 F

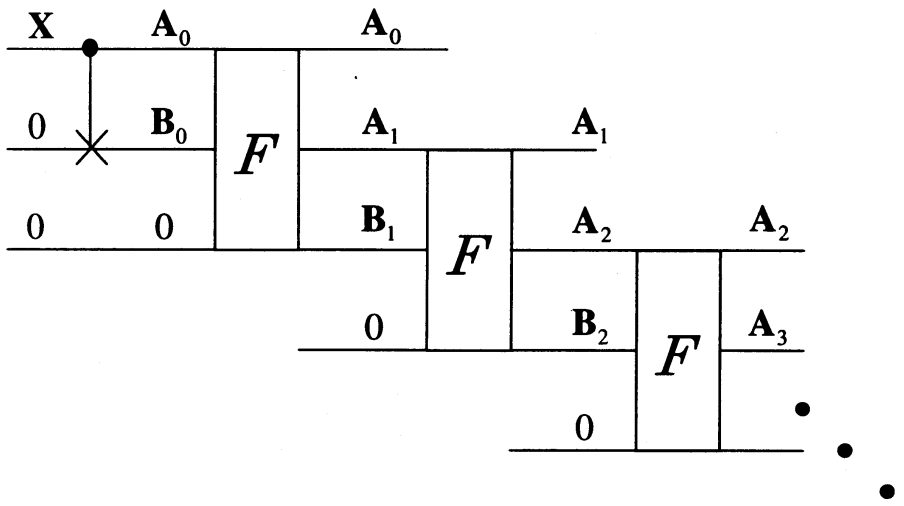

図 15: 漸化式に相当する F の繰り返し

の可逆性がここに表われている。また、$A_1, A_2, \ldots, A_{N-1}$ のビットは全体として入力"0" 出力"0" の過程の途中に表われているため、(古典的) プログラミングにおける、テンポラリ変数と見なすことができる。さらに大きな N の回路では、このテンポラリ変数を再利用し、全体で用いるビット数を減らすことも可能である。

まとめると、図 16 の量子回路はユニタリー変換

$$|X\rangle \otimes |0\rangle \otimes |0\rangle \otimes |0\rangle \otimes |0\rangle \mapsto |X\rangle \otimes |0\rangle \otimes |0\rangle \otimes |0\rangle \otimes |T_\infty X\rangle$$

を表している。ここで入力状態の重ね合わせをあらわに書くと、

$$\left(\sum_{n=0}^{2^N-1} c_n |n\rangle\right) \otimes |0\rangle \quad \mapsto \quad \sum_{n=0}^{2^N-1} c_n |n\rangle \otimes |T_\infty n\rangle$$

となる (途中のテンポラリな空間は省略した)。これは入力の重ね合わせ状態それぞれについて出力の状態が対応しているため、entanglement 状態となっている (cf. 式 (11))。

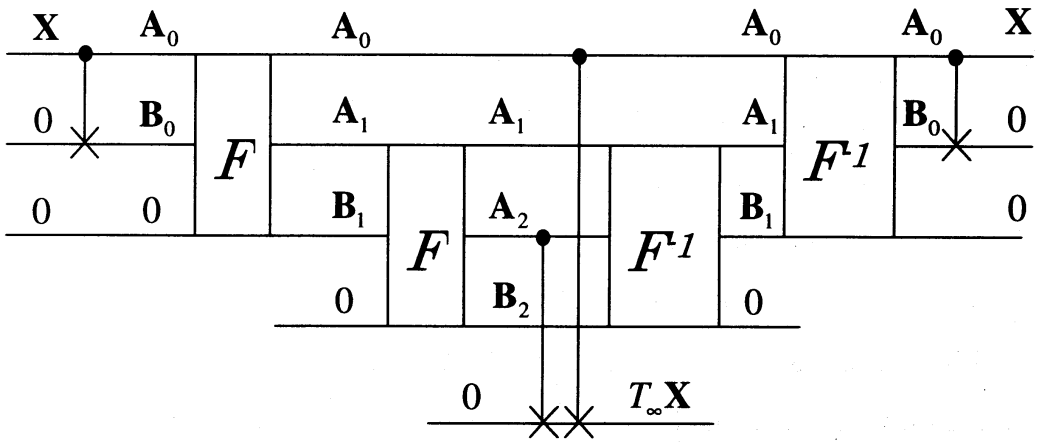

図 16: 漸化式の途中の $A_1, A_2, \ldots, A_{N-1}$ は 0 に戻すことができる (図は $N = 2$)

最後に計算量を見積もる。式(1)は、Nビットの入力に対しN段の漸化式で表わされている。その漸化式1段分の量子回路Fは入力サイズNに対して明らかに線形である(図14)。つまり回路長は$O(N^2)$であり、多項式時間の計算量クラスであることを示している。

6 考察と課題

本研究では周期的境界条件のもと有限な箱玉系をまず導入し、可解格子模型の立場から正当化できることを示した。このpBBSは、その周期性などといった未だよくわからない問題も含み、有限系ゆえの難しさがあると思われる。ここでは、回転シフトT_1のもとで初期状態をMöbius関数を用いて分類したが、さらなる組み合わせ論的な考察が必要である。逆に、組み合わせ論的な問題に逆超離散化が応用可能かもしれない。

箱玉系の量子回路としての実装が、論理漸化式を与えたことにより多項式時間$O(N^2)$で可能になったことは、新しい量子アルゴリズムの発見や具体的な問題への応用を目指す第一歩である。今回示した量子「回路」を量子「アルゴリズム」として応用するには、離散可積分系としてpBBSの具体的な構造をさらに調べなければならないだろう(cf. [18])。系のもつ保存量はアルゴリズムを構築する際の指針となり得るものであり、量子誤り訂正などに応用できる可能性もある。また(古典)可積分アルゴリズムとして、論理式(1)の逆超離散極限が$2N$乗根を求めるアルゴリズムに対応していることは特筆すべき点であろう。

残された課題は多いが、可積分系と量子コンピューティングの双方の掛け橋として寄与できれば幸いである。

7 謝辞

本研究は時弘哲治教授との共同研究であり、ソリトン理論と可解格子模型の関係などについて教えていただきました。また、薩摩順吉教授および両研究室の方々にも議論頂きました。この場を借りて御礼申し上げます。

参考文献

[1] P. W. Shor, Proceedings of the 35th Annual IEEE Symposium of Foundations of Computer Science(1994).

[2] P. Benioff, *Phys. Rev. Lett.* **48**, 1581(1982).

[3] R. P. Feynman, *Feynman Lectures on Computation*, Addison-Wesley(1996).

[4] D. Deutsch, *Proc. R. Soc. Lond.* **A 400**, 97(1985).

[5] H. Nishimura and M. Ozawa, quant-ph/9906095(1999).

[6] Y. Nakamura and A. Mukaihira, *Phys. Lett.* **A 249**, 295(1998).

[7] A. Nagai, T. Tokihiro and J. Satsuma, *Math. Comp.* **67**, 1565(1998).

[8] D. Takahashi and J. Satsuma, *J. Phys. Soc. Jpn.* **59**, 3514(1990).

[9] T. Tokihiro, D. Takahashi, J. Matsukidaira and J. Satsuma, *Phys. Rev. Lett.* **76**, 3247(1996).

[10] T. Tokihiro, D. Takahashi and J. Matsukidaira, *J. Phys.* **A. 33**, 607(2000).

[11] M. Torii, D. Takahashi and J. Satsuma, Physica **D92**, 209 (1996).

[12] D. Takahashi and J. Matsukidaira, *J. Phys.* **A. 30**, 733(1997).

[13] A. Nakayashiki and Y. Yamada, *Selecta Mathematica, New Series* **30**, 547(1997).

[14] T. Tokihiro, in private talks

[15] R. J. Baxter, Exactly Solved Models in Statistical Mechanics, Academic Press (1982).

[16] H. Rutishauser, *Z. Angew. Math. Phys.* **5**, 233(1954).

[17] F. Harary and E. M. Palmer, *Graphical Enumeration*, Acadeic Press(1973).

[18] R. Jozsa, quant-ph/9707033(1997).

[19] A. Barenco, C. H. Bennett, R. Cleve, D. P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. A. Smolin and H. Weinfurter, *Phys, Rev.* **A 52**, 3457 (1995).

[20] R. R. Tucci, quant-ph/9902062.