# あるクラフトゲームの計算モデルについて

2015SE027 伊藤 雅浩 2015SE088 鵜野 高寛 2015SE099 全 子陽
指導教員：横山哲郎

## 1 はじめに

計算モデルの研究は何を計算することができ，何を計算することができないかを知るために行われてきた．計算モデルには，チューリング機械をはじめとし，多くの種類が存在する．計算モデルの 1 つの指標としてチューリング機械と同等の計算能力を持つものをチューリング完全な計算モデルであるという．

Minecraft というクラフトゲームも，一種の計算モデルとして扱うことができる．Minecraft の計算モデルの計算能力は，チューリング機械と同じ計算能力を持っているという主張がある．これは「Minecraft はチューリング完全となった」（原文：“minecraft is now turing complete”）と，その当時 Minecraft の開発者の一人であった Markus Alexej Persson が 2010 年に発言しているからである [1].

また，多くの人が Minecraft 上でチューリング機械であるというものを作成し，WEB 上で公開をしている．チューリング機械は，マス目に分割され左右に無限長のテープ，有限制御部，テープ上の記号を読み書きするためのヘッドから構成される [3]．しかし，web 上で公開されているチューリング機械を実装したとされているものは，Minecraft の計算モデルがチューリング完全となるための条件を明らかにしておらず，水源，水流の形式化，レッドストーン回路の形式化を行っていない．またレッドストーン回路において，完全系である条件も示されてないので本当にチューリング完全であると言えているかがわからないという問題がある．

また，Minecraft にはブロックの設置範囲の問題と Minecraft のフィールドにおける時間停止の問題がある．

本研究は，Minecraft の計算モデルがチューリング完全であるための条件を示し，一般化 Minecraft を定義してそのチューリング完全性を示す．また，Minecraft 上で作成した論理ゲートがどの程度の能力を持つかを示す．これによる期待される効果は，Minecraft がどの程度の計算能力を持っているかがわかり，計算モデルの計算能力を示すときの一種の指標とすることができる．

本稿では，Minecraft を 3 次元セル・オートマトンとして形式化した一般化 Minecraft を考え，任意のチューリング機械を模倣できるカウンタ機械を一般化 Minecraft 上で実装する．

## 2 関連研究

### 2.1 論理ゲート

論理ゲートは，論理的な演算・処理を行うゲートを構成するための基本要素である論理素子を，有限個結合することによって構成されるゲートである [3].

論理ゲートのうち，NAND，あるいは NOT と AND を組み合わせることによって，いかなる論理ゲートも構成できる．これを論理ゲートの集合における完全系という．

### 2.2 計算モデル

計算モデルとは，数学的に表現できる情報処理を機械上の操作と考えたモデルである．計算モデルの例として，チューリング機械，セル・オートマトン，ラムダ計算などがある．

### 2.3 計算モデル間のシミュレーション

計算モデル間のシミュレーションは，計算モデルの計算能力がどれほどであるか知るために行われてきた．これは，ある計算モデル A が他の計算モデル B をシミュレーションできれば，A は B と同等の計算能力を持っているということが判明するためである．計算モデル間のシミュレーションの例としては，カウンタ機械でチューリング機械のシミュレーションなどがある．

### 2.4 一般化ぷよぷよの NP 完全性

一般化したぷよぷよというゲームにおいて連鎖数判定問題が NP 完全であることが知られている [2].

### 2.5 チューリング機械

チューリング機械とは，Turing によって考案された計算モデルである．チューリング機械は，ます目に分割され左右に無限に伸びたテープ，有限制御部，およびテープ上の記号を読み書きするためのヘッドから構成される．

計算理論において，ある計算のメカニズムがチューリング機械と同じ計算能力をもつとき，その計算モデルはチューリング完全であるという．計算モデルにおいて，チューリング完全を示すためにはすでにチューリング完全とわかっている何かを模倣すれば良い．なぜならその何かを模倣することによって間接的にあらゆるチューリング機械の計算を模倣できるからである．

### 2.6 セル・オートマトン

セル・オートマトン (以下，CA) は，1950 年代に von Neumann が提案した計算モデルである．本稿では CA を，

$$A = (\mathbb{Z}^k, Q, (n_1, \ldots, n_m), f, \#) \tag{1}$$

として定める．ただし，$\mathbb{Z}$ はすべての整数の集合，$Q$ は各セルの取る内部状態の空でない有限集合，$(n_1, \ldots, n_m)$ は $(\mathbb{Z}^k)^m$ の要素で，近傍と呼ぶ．関数 $f : Q^m \to Q$ は，各セルの状態を定める局所関数である．$\# \in Q$ は静止状態と呼ばれ，$f(\#, \ldots, \#) = \#$ を満たす．$q_1, \ldots, q_m, q \in Q$ に対し関係 $f(q_1, \ldots, q_m) = q$ が成り立つとき，この関係を遷移規則と呼ぶ [3].

写像 $\alpha : \mathbb{Z}^k \to Q$ を $A$ の状相とする．集合 $Q$ 上の $k$ 次元状相の集合を $Conf(Q)$ で表す．$x \in \mathbb{Z}^k$ である任意のセルを用いて $\alpha(x)$ を座標 $x$ に位置するセルの状態を表すとする．すると $A$ の大域関数 $F : Conf(Q) \to Conf(Q)$

は次のようになる．

$$\forall \alpha \in Conf(Q), \forall x \in \mathbb{Z}^k :$$
$$F(\alpha)(x) = f(\alpha(x+n_1), \ldots, (x+n_m))$$

すなわち，状相 $F(\alpha)$ は現在の状相 $\alpha$ の全てのセルに局所関数 $f$ を用いることによって得られることを示している．

## 3 Minecraft

### 3.1 Minecraft とは

Minecraft は，Markus Persson と Mojang AB の社員によって作られ，2011 年に正式版がリリースされたサンドボックス型クラフトゲームである．2018 年 1 月に，売上本数は 1 億 4400 本になっており，世界中でプレイされているゲームである．

Minecraft ではゲーム開始時にフィールドが水ブロックや土ブロックなど様々な立方体のブロックで生成され，構築される．またフィールドは立方体のマス目に区切られており，プレイヤーが一定の区間進むたびに無限に広がっていき自由に歩き回ることができる．また，Minecraft はチュートリアルやゲームとしての明確な目標，こうしなければならないという制約は定められていない．ブロックで建築を楽しむことや，ブロックやアイテム集めの効率化を図るなどプレイヤー自身が目標を決めて楽しむことができる設計となっている．

Minecraft 上では自身を中心とする $16 \times 16$ マスをチャンクと呼び，自身から $19 \times 19$ チャンク以外は時間が止まる問題がある．このため，$19 \times 19$ チャンク以上の大きさのカウンタ機械は動作を止めてしまう問題がある．フィールドに存在するブロックには，透過ブロックと非透過ブロックの 2 種類が存在する．3.4 節で説明するレッドストーン回路のレッドストーンダストの段差における接続の場合に段差をブロックで遮り，回路の接続が切れなかったブロックを透過ブロック，切れた場合を非透過ブロックとする．

また，Minecraft は $[-29999984, 29999984] \times [0, 255] \times [-29999984, 29999984]$ の範囲にしかブロックを設置することができない．このため本稿ではブロックの設置できる範囲の制限，及び時間が停止する制限を緩和した一般化 Minecraft でチューリング完全を示す Minecraft 内に存在するものは，一定のルールに従っているものが多いため，本稿では 3 次元 CA として Minecraft の計算モデルを形式化する．

### 3.2 水源と水流の定義

Minecraft には水が存在し，それは水源と水流の 2 つの種類に分けられる．水源は周囲の座標にブロックがない場合，自身から周囲の座標に水流を生成することができ，水流は水源から 1 マス流れるごとに 8 分の 1 マスずつ高さが減少する．また，複数のマスから同じマスに水流が流れるとき，そのマスの水流は，複数の水流のうちの一番高さが高い水流を参照して高さを 8 分の 1 マス減少させる．また，Minecraft には水流と似た性質を持つ溶岩と溶岩流が存在し，定義を類推することができる．水源と水流を 3 次元 CA

$$C_{flow} = (B_{flow}, Q_{flow}, N_{flow}, f_{flow}, \#) \tag{2}$$

と定める．ここで，

$$B_{flow} = \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z}$$
$$Q_{flow} = \{1, \ldots, m+1\} \cup \{\#, b, s, c\}$$
$$N_{flow} = ((0,0,0),(0,0,1),(1,0,0),(0,0,-1),$$
$$(-1,0,0),(0,1,0),(0,-1,0))$$

$$f_{flow}(q_0, q_1, q_2, q_3, q_4, q_5, q_6) =$$
**if** $q_0 = \#$ **then**
$$\begin{cases} \# & q_5 \neq s \\ s & q_5 = s \end{cases}$$
**else**
$$\begin{cases} b & q_0 = b \\ c & q_0 = c \wedge \max\limits_{1 \leq i \leq 4} \{q_i\} = 0 \\ s & q_0 = s \wedge q_6 \neq \# \\ \# & (q_0 = s \wedge q_6 = \#) \ \vee (q_0 = c \ \wedge \max\limits_{1 \leq i \leq 4} \{q_i\} \geq 2) \\ i & q_0 \leq m \wedge \neg(1 \leq q_5 \leq m+1) \ \wedge \max\limits_{1 \leq j \leq 4} \{q_j\} = i+1 \\ m & 1 \leq q_5 \leq m+1 \\ m+1 & (q_i = q_j = m+1 \wedge i \neq j \wedge 1 \leq i, j \leq 4) \vee q_0 = m+1 \end{cases}$$

である．$Q_{flow}$ は水の高さとマスに存在しているブロックの集合，$N_{flow}$ は近傍である．$\#$ はブロックが配置されていない空気，$b$ はブロック，$s$ は砂，$c$ はカーペットである．また $m$ は水流の最大の高さを表しておりここでは $m = 8$，$m+1$ は高さ $m$ の水源として扱う．

### 3.3 水によるゲート

水流とブロックによって NOT, OR, AND ゲートなどの論理ゲートを作成することは可能である．ここで，水とブロックによって構成された NAND ゲートの模式図を図 1 に示す．$W$ は水源，$w$ は水源から生成された水流，$b$ はブロック，$s$ は砂ブロック，$c$ はカーペットである．入力は A, B で出力は O である．また，図では見えないが，砂ブロックの 1 マス下にはカーペット，2 マス下には水源があるとする．砂ブロックは，1 マス下が水流やブロックが存在しないマスの場合 1 マス下に移動する．看板は水流や溶岩流を遮るという性質を持つ．カーペットは水流が流れてくると自身が破壊されるという性質を持つ．

このようにして NAND ゲートは作成できるが，レッドストーン回路を用いずに水を上方に上げる方法は存在せず，$0 \leq y \leq 255$ の高さの制限も存在する．よって水とブロックのみで構成されるゲートは高さの問題がなく，ゲートの存在するチャンクの時間が止まっていなければ，完全系となる．

### 3.4 レッドストーン回路

レッドストーン (以下，RS) 回路とは，Minecraft のフィールド上で，装置に機械的な動作をさせるための建築物である．RS 回路を構成する部品は，動力を供給する動力部品，動力を受け渡す伝達部品，動力の有無によって自身が作用する機械部品がある．

RS 回路の動力部品には RS トーチが存在する．RS トーチは設置されている向き以外に動力を供給する．また，設置されている非透過ブロックに動力が伝わっている間，動

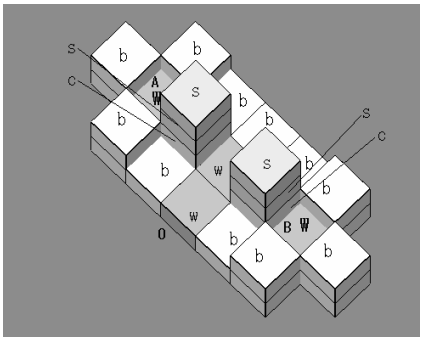


図 1 水流とブロックで構成された NAND ゲート

力を供給しなくなる．これは，NOT ゲートとして振舞っていると考えることができる．伝達部品には，RS ダスト，RS リピータが存在する．RS ダストは．また動力を伝えることができ，回路にける導線の役割を果たしている．しかし動力源から 15 マス先までしか動力を伝えず、動力が弱まっていく制限がある．RS リピータは，指向性のある部品で，回路の延長，増幅機構も備えており，これを使用することによって RS ダストの距離における制限は無視できる．また，遅延回路としても作用させることができ，2 つ使用することにより動力の一時保持も可能となっている．これよりこの機構を用いることで順序回路を構成することができる．

### 3.5 RS 回路の形式化

RS 回路は，3 次元 CA として表すことができる．2.6 節より，

$$R = (\mathbb{Z} \times \{0, \ldots, 255\} \times \mathbb{Z}, (Q \times Q'), N, f, (0, 0)) \quad (3)$$

と定義できる．ここで，$Q$ は動力の状態の集合で $Q = \{-1, \ldots, 16\}$, RS トーチに影響を及ぼす状態を-1，動力が伝達していない状態を 0，動力が伝達している状態を $1, \ldots, 16$ とする．$Q'$ はブロックの種類の状態の集合で $Q' = \{b_{nt}, b_t, b_{tn}, b_{te}, b_{ts}, b_{tw}, d, t_n, t_e, t_s, t_w, t_d, rp_n, rp_e, rp_s, rp_w\}$ ，$b_{nt}$ は非透過ブロック，$b_t$ は透過ブロック，$b_{tn}, b_{te}, b_{ts}, b_{tw}$ はそれぞれ下と北，下と東，下と南，下と西向きにしか動力を伝えない透過ブロック，$d$ は RS ダスト，$t_n, t_e, t_s, t_w, t_d$ はそれぞれ北，東，南，西，上向きの RS トーチ，$rp_n, rp_e, rp_s, rp_w$ はそれぞれ北，南，東，西向きの RS リピータとする．ここで，ブロックの種類を透過ブロックと非透過ブロックの 2 種類に分けるのは RS 回路の RS ダストの段差における接続の表現を記述するためである．

近傍は，$N = ((0,0,0),(0,0,1),(1,0,0),(0,0,-1),(-1,0,0),(0,1,0),(0,-1,0))$，局所関数は図 2 である．

また，RS 回路を用いて NAND ゲートを作成することができる．図 3 に RS 回路で作成した NAND ゲートの模式図を示す．ここで，$b_{nt}$ は非透過ブロック，$d$ は RS ダスト，$t_d$ は下付きの RS トーチである．これは，両方の入力 A,B が 1 の時に 2 つの RS トーチが NOT ゲートの振る舞いをするので動力を供給しなくなり，出力 O が 0 となる．これより，RS 回路は，回路の存在するチャンクの時間が止まっていなければ，完全系となる．

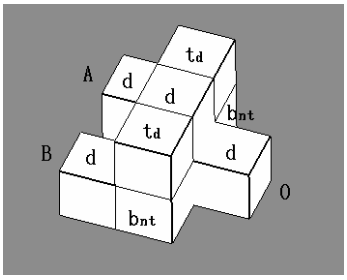


図 3 RS 回路で構成された NAND ゲート

## 4 カウンタ機械

カウンタ機械は計算機のモデル理論モデルの 1 つであり，チューリング完全であることが知られている [3].

カウンタ機械はマス目に分割された右方に無限長のテープを持ち，任意の大きさの非負整数を 1 つ蓄えられるような有限個のカウンタと，それにアクセスできる有限制御部によって構成されている．有限制御部が各カウンタに対して実行できる演算，操作はカウンタが保持する数が 0 か否かの判定と，その数を 1 増やす，あるいは 1 を減らすという演算だけである．i 番目のテープヘッドが左から $n$ 番目のマス目にあるとき i 番目のカウンタ機械が $n$ を保持しているとする．各テープヘッドは記号を書き換えることができず，カウンタの値が 0 なのか正なのかに応じて次の時刻の内部状態とカウンタの値を 1 増やすか，1 減らすか増減なしかを決定し，実行することで計算を行う．

## 5 Minecraft でのカウンタ機械の構成

一般化 Minecraft の計算モデルがチューリング完全であることを示すには万能カウンタ機械，すなわち任意のチューリング機械を模倣するカウンタ機械を構成する必要がある．

Minecraft 上で任意のチューリング機械を模倣できるカウンタ機械を構成するには，3.4 節，4 章より，カウンタ機械をマスの時間が停止していない，無限長のテープをもつカウンタ機械が作成可能であるという条件がある．また，カウンタが 10 個必要でチューリング機械の状態 $q_i$ における動作を行う方法と 5 項組列が格納されているカウンタから 5 項組を取り出し別のカウンタに格納する方法が必要である．

次に、Minecraft におけるカウンタ機械の構成を示す．Minecraft でカウンタ機械を構成するにはカウンタと有限制御部が必要である．カウンタを構成するには，XOR ゲート，AND ゲート，NOT ゲートを用いる．カウンタを図 4 に示す．図 4 の一番左には XOR ゲートを用意し，その右側には上下で組の AND ゲートを無数に用意する．上側のそれぞれの AND ゲートの入力の 1 つの手前には RS リピータが 2 つ存在し動力を保持できるように設置されている．この保持は，カウンタに加算，または減算の命令が来た時に動力が伝わっている間，動力保持の状態を解除する．AND ゲートの入力のもう 1 つには，入力

$$
\begin{aligned}
&\forall q_0,q_1,q_2,q_3,q_4,q_5,q_6 \in Q, \forall q'_0,q'_1,q'_2,q'_3,q'_4,q'_5,q'_6 \in Q' : \\
&f(q,q') = \\
&\begin{cases}
(-1,q'_0) \quad q'_0 = b_{nt} \wedge (1 \le q_i \le 15 \wedge q'_i = d \wedge 1 \le i \le 5) \\
(0,q'_0) \quad q_i = 0 \text{ for } i = 0,\ldots,6 \vee ((1 \le q_0 \le 15) > \max\{q_1,q_2,q_3,q_4,q_5,q_6\} \wedge q'_0 = d) \\
\qquad \vee (q'_0 \in \{t_d,t_n,t_e,t_s,t_w\} \wedge q'_i = b_{nt} \wedge (q_i = 16 \vee q_i = -1) \wedge i = \begin{cases} 6 & q'_0 = t_d \\ 3 & q'_0 = t_n \\ 1 & q'_0 = t_s \end{cases} \quad \begin{matrix} 4 & q'_0 = t_e \\ 2 & q'_0 = t_w \end{matrix} \quad ) \\
(i,q'_0) \quad q'_0 = d \wedge \max\{q_1,q_2,q_3,q_4,q_5,q_6\} = i+1 \wedge 0 \le i \le 15 \\
\qquad \vee (q'_0 \in \{b_{tn},b_{te},b_{ts},b_{tw}\} \wedge 1 \le i \le 16 \wedge ((q_j = i \wedge j = \begin{cases} 3 & q'_0 = b_{tn} \\ 1 & q'_0 = b_{ts} \end{cases} \quad \begin{matrix} 4 & q'_0 = b_{te} \\ 2 & q'_0 = b_{tw} \end{matrix} \quad ) \vee q_6 = i) \wedge q'_6 = d) \\
\qquad \vee (q'_0 \in \{rp_n,rp_e,rp_s,rp_w\} \wedge q_j = i \wedge j = \begin{cases} 3 & q'_0 = rp_n \\ 1 & q'_0 = rp_s \end{cases} \quad \begin{matrix} 4 & q'_0 = rp_e \\ 2 & q'_0 = rp_w \end{matrix} \quad \wedge i = \begin{cases} 16 & 1 \le q_j \le 16 \\ 0 & q_j = 0 \end{cases} \quad ) \\
\qquad \vee (q'_0 \in \{rp_n,rp_e,rp_s,rp_w\} \wedge q_0 = i \wedge j = \begin{cases} 1 & q'_0 = rp_n \vee q'_0 = rp_s \\ 2 & q'_0 = rp_e \vee q'_0 = rp_w \end{cases} \\
\qquad\quad \wedge \begin{cases} (q'_2 = rp_w \wedge q_2 = 16) \vee (q'_4 = rp_e \wedge q_4 = 16) & j = 1 \\ (q'_1 = rp_s \wedge q_1 = 16) \vee (q'_3 = rp_n \wedge q_3 = 16) & j = 2 \end{cases} \quad \wedge i \in \{0,16\}) \\
\qquad \vee (q'_0 = b_{nt} \wedge q'_6 \in \{t_d,t_n,t_e,t_s,t_w\} \wedge q_6 = i \wedge i \in \{0,16\}) \\
(16,q'_0) \quad q'_0 = b_{nt} \wedge q'_i = rp \wedge 1 \le i \le 4 \wedge q_i = 16 \wedge rp = \begin{cases} rp_n & i = 3 \\ rp_s & i = 1 \end{cases} \quad \begin{matrix} rp_e & i = 4 \\ rp_w & i = 2 \end{matrix}
\end{cases}
\end{aligned}
$$

図 2　3 次元 CA として形式化した RS 回路の局所関数

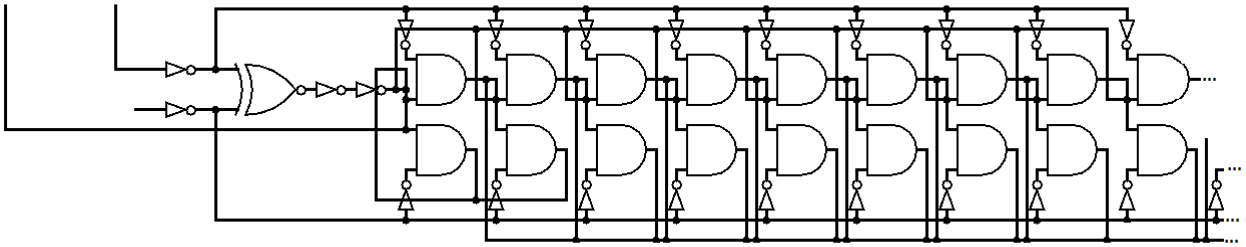

図 4　Minecraft 上で実装したカウンタの回路図

の部分から動力がつながっており AND ゲートの直前に NOT ゲートが存在する．これより XOR ゲートの入力のどちらかに動力がきたかで上側と下側のいずれかの AND ゲートの入力の 1 つに動力が伝わるかが決まる．上側の AND ゲートの出力が 1 になった場合は，その AND ゲートの 1 つ右側にある AND ゲートの手前の RS リピータに動力が保持される．また，下側の AND ゲートの出力が 1 になった場合は，その AND ゲートの 1 つ左の AND ゲートの手前の RS リピータに動力が保持される．

これより，RS リピータの位置を左側から $0,1,2,\ldots$，とすると RS リピータが動力を保持している場所によって格納している値を確認できる．カウンタの 2 つの入力の上側に動力が伝わった場合，カウンタの値を 1 加え，下側に動力が伝わった場合，カウンタの値を 1 減らす．

また，一番左の AND ゲートの手前に存在する RS リピータの先から分岐させ，その先に AND ゲートを作成するとカウンタ機械の 0 か否かを判定する機能を実装できる．これより有限制御部による演算のすべてが行える．この構成方法で Minecraft 上で実装することができる．カウンタ機械は，RS 回路を用いることにより Minecraft 上で構成可能となる．

## 6　おわりに

本研究では，Minecraft の計算モデルがチューリング完全となるための条件を示し，水流，RS 回路の形式化を行い，ゲートの集合が完全系である条件を示した．また，任意のチューリング機械を模倣することができるカウンタ機械を構成的に示し，一般化 Minecraft の計算モデルがチューリング完全であることを構成的に示した．

## 参考文献


[1] Markus Alexej Persson: Notch さんのツイート: ⟨@hideous_ minecraft is now turing complete⟩，Twitter（オンライン）⟨https://twitter.com/notch/status/17644112984⟩（参照 2018-06-22）．

[2] 松金輝久，武永康彦: 組合せ最適化問題としてのぷよぷよの連鎖数判定問題，電子情報通信学会論文誌 D,Vol.J89D,No.3,pp.405413 (2006)．

[3] 森田憲一: 可逆計算，近代科学社 (2012)．

[4] Marvin L.Minsky，金山 裕 (訳)：“計算機の数学的理論”，近代科学社 (1970)．