Конструирование изображений клеточными автоматами

Е. Е. Титова

В работе рассматривается задача конструирования изображений клеточными автоматами на прямоугольном экране. Показано, что для любого изображения необходимо и достаточно, чтобы клеточный автомат имел 3 состояния. Получены оценки времени конструирования изображений в зависимости от числа состояний клеточного автомата.

Введение

Рассмотрим в двумерном пространстве прямоугольник размера $n \times m$ клеток, $m \geqslant n \geqslant 3$, $m, n \in \mathbb{N}$. В каждую клетку прямоугольника поместим по одному экземпляру одного и того же конечного автомата. К входам этого автомата присоединим выходы автоматов, стоящих в четырех соседних с ним клетках. Выходом автомата в каждый момент времени является его состояние в этот момент времени. Автоматы, стоящие в первой и n-й строках и в первом и m-м столбцах будем называть граничными автоматами. Для этих автоматов определены не все входы. Поэтому будем считать, что у автоматов, стоящих в m-м столбце правый вход всегда нулевой, аналогично у автоматов, стоящих в n-й строке нижний вход всегда нулевой. Неопределенные входы автоматов первой строки и первого столбца будем называть свободными входами. Итак, автоматы, стоящие в клетках прямоугольника, будем называть клеточными автоматами. Конфигурацию из нулевых и единичных состояний клеточных автоматов назовем изображением, если эту конфигурацию можно удерживать

сколь угодно долго, подавая на свободные входы автоматов нулевые значения. Задача состоит в формировании таких последовательностей для свободных входов клеточных автоматов, чтобы через какое-то время на экране сформировалось изображение, причем положение нулей и единиц в нем заранее задано.

В работе показано, что для любого изображения необходимо и достаточно, чтобы клеточный автомат имел 3 состояния. Получены оценки времени конструирования изображений в зависимости от числа состояний клеточного автомата.

1. Основные понятия и формулировка результатов

Пусть имеется конечный автомат $\mathcal{A}$ с четырьмя входами. $E_q = \{0, 1, \ldots, q-1\}$ — множество состояний автомата, $\varphi(q, l, r, u, d)$ — функция переходов состояний автомата, где q — текущее состояние, l, r, u, d принимают значения из E_q, причем имеет место свойство $\varphi(0, 0, 0, 0, 0) = 0$. Функция выхода автомата совпадает с функцией переходов состояний. Описанный автомат будем называть *клеточным* автоматом. Пусть имеется прямоугольник размера $n \times m$, $m \geqslant n \geqslant 3$, $m, n \in \mathbb{N}$. В каждую клетку прямоугольника поместим по одному экземпляру одного и того же клеточного автомата $\mathcal{A}$. К входам этого автомата присоединим выходы автоматов, стоящих в четырех соседних с ним клетках, то есть у него имеется *левый* вход l, *правый* вход r, *верхний* вход u и *нижний* вход d соответственно и q — текущее состояние автомата. Выходом автомата является его состояние в этот момент времени. Автоматы, стоящие в первой и n-й строках и в первом и m-м столбцах будем называть *граничными автоматами*. Для этих автоматов определены не все входы. Будем считать, что у автоматов, стоящих в m-м столбце правый вход всегда нулевой, и у автоматов, стоящих в n-й строке нижний вход всегда нулевой. Неопределенные входы автоматов первой строки и первого столбца будем называть *свободными входами*. Описанную конструкцию назовем (n, m)-экраном, то есть (n, m)-*экраном* S назовем тройку

$S = <\mathcal{A}, n, m>$, где $\mathcal{A}$ — клеточный автомат, n и m — соответственно количество строк и столбцов прямоугольника.

Состояния 0 и 1 клеточных автоматов будем называть *метками*. Если каждый клеточный автомат экрана находится в нулевом или единичном состоянии, то такую конфигурацию состояний экрана будем называть *черно-белой конфигурацией*. Черно-белую конфигурацию назовем *изображением*, если эту конфигурацию можно удерживать сколь угодно долго, подавая на свободные входы автоматов нулевые значения. *Кодом* K изображения назовем матрицу $n \times m$, состоящую из нулей и единиц. Скажем, что изображение $\Im_K$ *соответствует данному коду* K, если положение нулей и единиц в изображении и в коде совпадают.

Пусть также имеется *внешний* автономный автомат $\mathcal{A}_e$ с множеством состояний $E_{q'} = \{0, 1, \ldots, q' - 1\}$, который генерирует последовательности входных символов из множества E_q для свободных входов клеточных автоматов. То есть выход внешнего автомата — два вектора: $(a_1, \ldots, a_n)$ и $(b_1, \ldots, b_m)$, где $a_i, b_j \in E_q$, $i = 1, \ldots, n$, $j = 1, \ldots, m$. В начальный момент времени внешний автомат находится в начальном, нулевом состоянии, а на экране имеется черно-белое изображение (либо экран из одних нулей). Задача состоит в построении такого клеточного автомата, что для любого кода K можно построить такой внешний автомат, что при подаче выходов внешнего автомата на входы экрана, на экране генерируется изображение $\Im_K$, соответствующее коду K. Обозначим $\Im(n, m)$ — множество изображений размера $(n \times m)$, соответствующих всем возможным кодам.

Генератором G назовем пару $G = <\mathcal{A}_e, S>$, где $\mathcal{A}_e$ — внешний автомат, S — экран. Если $\mathcal{A}_e$ — автономный автомат с $n + m$ выходами, S — (n, m)-экран с черно-белым начальным состоянием, то скажем, что генератор $<\mathcal{A}_e, S>$ формирует изображение с кодом K, если через некоторое время на экране появляется изображение, соответствующее коду K. Скажем, что экран $S = <\mathcal{A}, n, m>$ — *универсальный*, если для любого кода K существует внешний автомат $\mathcal{A}_e^K$, такой что генератор $G = <\mathcal{A}_e^K, S>$ формирует изображение $\Im_K$, соответствующее коду K. Множество всех универсальных (n, m)-экранов обозначим через $\mathcal{U}(n, m)$. Далее будет показано, что $\mathcal{U}(n, m) \neq \varnothing$ при

любых $n, m \geqslant 3$, $n, m \in \mathbb{N}$. Если $S = <\mathcal{A}, n, m>$ — экран, то через $Q(S)$ обозначим число состояний клеточного автомата $\mathcal{A}$. Обозначим

$$Q(n, m) = \min_{S \in \mathcal{U}(n,m)} Q(S).$$

Теорема 1. *Если $m \geqslant n \geqslant 3$, $n, m \in \mathbb{N}$, то $Q(n, m) = 3$.*

Для генератора $<\mathcal{A}_e, S>$ через $T(\mathcal{A}_e, S)$ обозначим момент времени, после которого конфигурация экрана не изменяется. Считаем, что $T(\mathcal{A}_e, S) = \infty$, если стабилизация не наступает. Если генератор $<\mathcal{A}_e, S>$ формирует изображение $\Im_K$, соответствующее коду K, то $T(\mathcal{A}_e, S)$ — время формирования изображения $\Im_K$.

Пусть S — универсальный экран, $\Im$ — изображение. Через $\mathcal{G}(S, \Im)$ обозначим множество генераторов $<\mathcal{A}_e, S>$, формирующих изображение $\Im$. Обозначим

$$T(S, \Im) = \min_{<\mathcal{A}_e, S> \in \mathcal{G}(S,\Im)} T(\mathcal{A}, S),$$
$$T(S, n, m) = \max_{\Im \in \mathfrak{I}(n,m)} T(S, \Im),$$
$$T(n, m) = \min_{S \in \mathcal{U}(n,m)} T(S, n, m),$$
$$T(n, m, q) = \min_{S \in \mathcal{U}(n,m), Q(S) \leqslant q} T(S, n, m).$$

Теорема 2. *Если $m \geqslant n \geqslant 3$, $n, m \in \mathbb{N}$, то*

$$T(n, m) = n.$$

Теорема 3. *Если $m \geqslant n \geqslant 3$, $n, m, k \in \mathbb{N}$, то*

$$T(n, m, 2^{k+1} - 1) \leqslant (m - k + 2) \cdot]n/k[+ k - 1.$$

2. Точная оценка числа состояний клеточных автоматов

Лемма 1. *Если $m \geqslant n \geqslant 3$, $n, m \in \mathbb{N}$, то $Q(n, m) > 2$.*

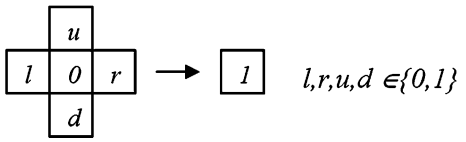

Рис. 1. Конфигурация входов клеточного автомата, переводящая состояние 0 в состояние 1.

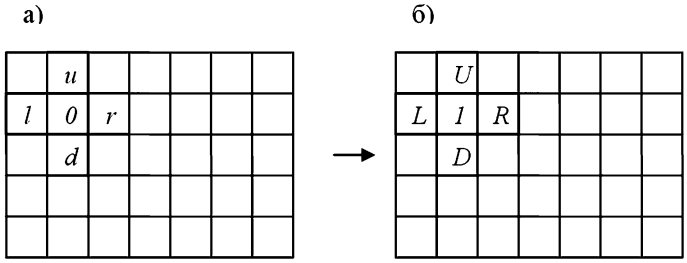

Рис. 2. Пример изображения, которое сохраняется только один такт.

Доказательство. Будем доказывать от противного. Пусть $S = <\mathcal{A}, n, m>$ — универсальный (n, m)-экран и клеточный автомат имеет не больше 2 состояний. Если клеточный автомат имеет только одно состояние, то конфигурация на экране никогда не изменится, и мы не сможем получить никакое другое изображение. Пусть клеточный автомат имеет 2 состояния из множества $E_2 = \{0, 1\}$. Так как экран универсальный, то для любого изображения существует внешний автомат $\mathcal{A}_e$ такой, что генератор $\mathcal{P} = <\mathcal{A}_e, S>$ формирует это изображение. То есть это условие выполнено и для изображения, содержащего одновременно и нулевые и единичные состояния клеточных автоматов. Значит существует конфигурация, переводящая состояние 0 клеточного автомата в состояние 1, то есть существуют такие $l, r, u, d \in E_2$, что $\varphi(0, l, r, u, d) = 1$, причем не все l, r, u и d равны нулю (см. рисунок 1). Но тогда мы не сможем получить изображение, приведенное на рисунке 2 а), так как оно сохраняется только один такт и переходит в конфигурацию 2 б). Это противоречит с тем, что экран универсальный. Лемма доказана.

Лемма 2. *Существует универсальный экран* $S = <\mathcal{A}, n, m>$, *такой что* $Q(S) = 3$.

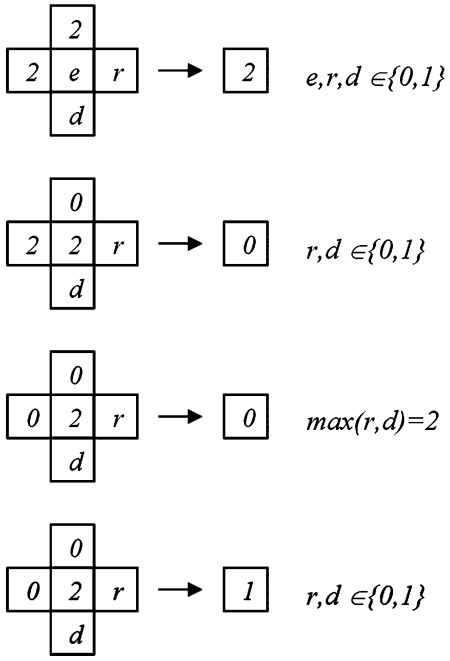

Рис. 3. Переходы состояний клеточного автомата с тремя состояниями.

Доказательство. Опишем клеточный автомат. У него три состояния из $E_3 = \{0, 1, 2\}$. Опишем функцию переходов автомата.

$\varphi(0, 2, r, 2, d) = 2$, где $r, d \in E_2$, в остальных случаях $\varphi(0, l, r, u, d) = 0$.

$\varphi(1, 2, r, 2, d) = 2$, если $r, d \in E_2$, в остальных случаях $\varphi(1, l, r, u, d) = 1$.

$\varphi(2, 0, r, 0, d) = 0$, если хотя бы одна из r и d равна 2, вторая принимает любое значение из E_3;

$\varphi(2, 0, r, 0, d) = 1$, если $r, d \in E_2$; $\varphi(2, 2, r, 0, d) = 0$, если $r, d \in E_2$, в остальных случаях $\varphi(2, l, r, u, d) = 2$.

На рисунке 3 показано, как меняются состояния клеточного автомата в зависимости от конфигурации входных элементов.

Опишем внешний автомат для данного двоичного кода изображения K. Он подает на входы клеточных автоматов такие последовательности, что точки рисуются по одной начиная с нижней строки справа налево. Причем, если в коде есть единица с координатами (n, m), то автомат сразу начинает генерировать последовательности для построения точек, в противном случае сначала очищает экран, то есть переводит все клеточные автоматы экрана в нулевое состояние, а затем уже конструирует последовательности для построения новых точек.

Опишем последовательности, которые будет генерировать внешний автомат чтобы очистить экран от единичек.

В первый такт внешний автомат подает векторы $(2, 0, \ldots, 0)_n$ и $(2, 0, \ldots, 0)_m$ (индексом внизу будем обозначать длину вектора); во второй такт — векторы $(2, 2, 0, \ldots, 0)_n$ и $(2, 2, 0, \ldots, 0)_m$; в третий — векторы $(0, 0, 2, 0, \ldots, 0)_n$ и $(0, 0, 2, 0, \ldots, 0)_m$ и так далее до n-го такта он подает векторы, у которых в момент k $(k \leqslant n)$ на k-м месте стоит двойка, на остальных местах — нули. В $(n+1)$-й такт — $(0, \ldots, 0, 2)_n$ и $(0, \ldots, 0, 2, 0, \ldots, 0)_m$, где 2 стоит на $(m+1)$-м месте. Дальше начиная с $(n+2)$-го и до m-го такта он подает первый вектор — нулевой, а второй такой, что в момент k $(i < k \leqslant m)$ на k-м месте стоит двойка, на остальных местах — нули, то есть $(0, \ldots, 0)_n$ и $(0, \ldots, 0, 2, 0, \ldots, 0)_m$. Затем автомат начинает генерировать последовательности для построения точек изображения.

Поясним алгоритм очистки экрана. Мы посылаем волну из двоек (она состоит из двух соседних диагоналей), причем в нижней строке этой волны ставим три двойки. Тогда, пройдя до конца экрана, все двойки кроме двух нижних исчезнут, а в следующий такт оставшиеся две двойки тоже исчезнут (так задана функция переходов состояний клеточных автоматов). Таким образом, все клеточные автоматы экрана окажутся в нулевом состоянии. На рисунке 4 приведен пример очистки экрана.

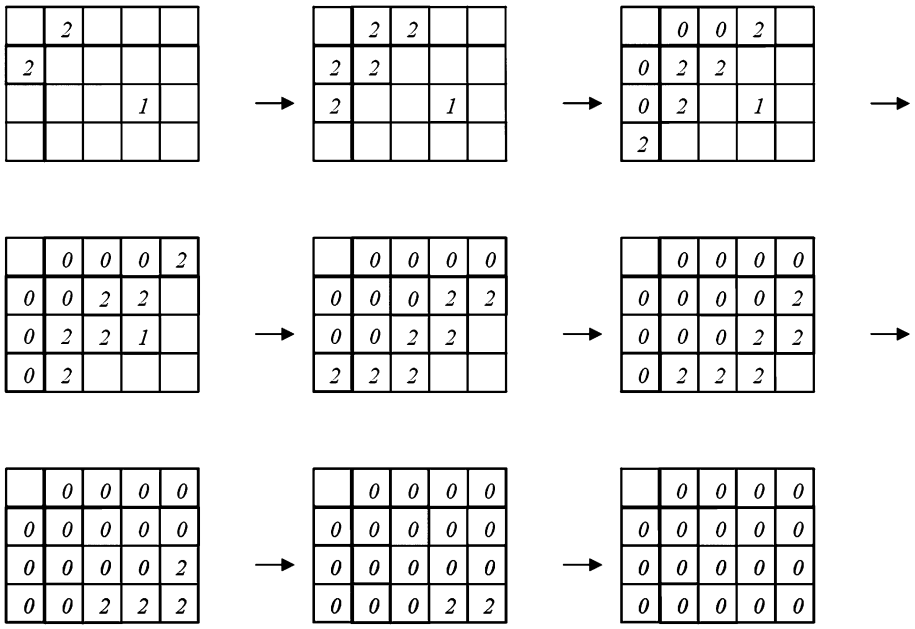

Рис. 4. Пример очистки экрана.

Опишем теперь последовательности, которые будет генерировать внешний автомат чтобы нарисовать точку с координатами (i, j) (то есть точку, стоящую в i-й сверху строке и j-м слева направо столбце).

В первый такт автомат выдает векторы $(2, 0, \ldots, 0)_n$ и $(2, 0, \ldots, 0)_m$.

Если $(i, j) = (1, 1)$, то во 2-й такт он выдает два нулевых вектора.

Если $0 < i \leqslant j$ и $(i, j) \neq (1, 1)$, во второй такт автомат выдает векторы $(2, 2, 0, \ldots, 0)_n$ и $(2, 2, 0, \ldots, 0)_m$; в третий — векторы $(0, 0, 2, 0, \ldots, 0)_n$ и $(0, 0, 2, 0, \ldots, 0)_m$ и так далее до i-го такта он подает векторы, у которых в момент k $(k \leqslant i)$ на k-м месте стоит двойка, на остальных местах — нули. Дальше начиная с $(i + 1)$-го и до j-го такта он подает первый вектор — нулевой, а второй такой, что в момент k $(i < k \leqslant j)$ на k-м месте стоит двойка, на остальных местах — нули, то есть $(0, \ldots, 0)_n$ и $(0, \ldots, 0, 2, 0, \ldots, 0)_m$. Затем автомат начинает генерировать последовательности для следующей точки.

Если $0 < j < i$ и $(i, j) \neq (1, 1)$, то во второй такт автомат выдает векторы $(2, 2, 0, \ldots, 0)_n$ и $(2, 2, 0, \ldots, 0)_m$; в третий — векторы $(0, 0, 2, 0, \ldots, 0)_n$ и $(0, 0, 2, 0, \ldots, 0)_m$ и так далее до j-го такта он подает векторы, у которых в момент k $(k \leqslant i)$ на k-м месте стоит двойка, на остальных местах — нули. Дальше начиная с $(j + 1)$-го и до i-го такта он подает второй вектор — нулевой, а первый такой, что в момент k $(j < k \leqslant i)$ на k-м месте стоит двойка, на остальных местах — нули, то есть $(0, \ldots, 0, 2, 0, \ldots, 0)_n$ и $(0, \ldots, 0)_m$.

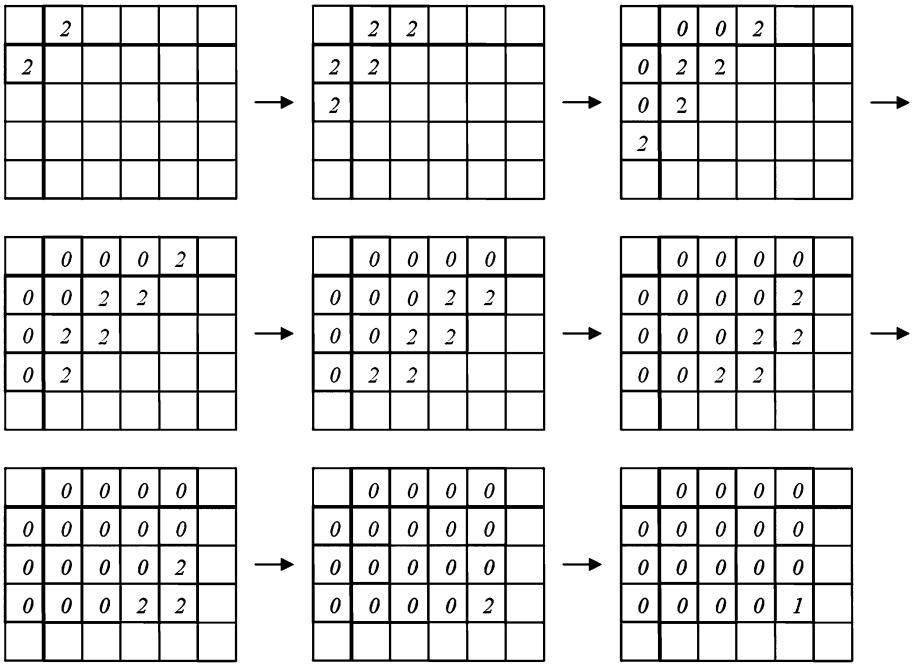

Рис. 5. Пример конструирования одной точки.

На рисунке 5 приведен пример построения точки с координатами $(3, 4)$. Заметим, что при построении точки с координатами (i, j) изменения происходят только в прямоугольнике $i \times j$, а за его пределами стоят только нули и единицы. Тогда из задания функции переходов φ видно, что уже построенные точки не мешают формированию новой и наоборот, построение новой точки не влияет на точки, уже суще-

ствующие на экране. Заметим также, что для построения точек мы использовали волну из двоек, причем двигаясь вперед, она оставляет за собой автоматы в нулевом состоянии. Поэтому, если мы рисуем точку с координатами (n, m), то волна проходит через весь экран, и конструируя точку, одновременно и очищает всю остальную часть экрана.

Таким образом можно представить внешний автомат как $n + m$ «маленьких» автоматов, каждый из которых генерирует 1 бит для выходных векторов. Эти автоматы разбиты на 2 группы: в одной n, в другой — m автоматов соответственно. В каждой группе автоматы выстроены цепочкой от 1-го до n-го и от 1-го до m-го соответственно. У этих автоматов есть счетчики, значения которых в начальном состоянии равны 0. Внешний автомат подает первым двум автоматам значения счетчика, равного соответствующей координате точки, которую конструирует автомат. В каждый момент времени каждый из автоматов (кроме первого и последнего) получает на вход значение счетчика от предыдущего автомата. Если оно равно 0, выдает нулевой бит в вектор и подает 0 на вход следующему автомату. Иначе выдает 2 в вектор, уменьшает значение счетчика на 1 и передает это новое значение счетчика на вход следующему автомату. Первый автомат ведет себя так же, но подает 2 в выходной вектор два такта подряд. Отличие последнего автомата в том, что у него только один выход — бит вектора, и нет выхода для следующего автомата, так как он сам последний. То есть первые автоматы имеют по 2 состояния, остальные — по одному.

Кроме того есть $p \leqslant mn$ состояний внешнего автомата, каждое из которых соответствует единичному биту из двоичного кода изображения (справа налево в строке начиная с нижней). Каждому из этих состояний соответствует пара чисел (i, j) — это координаты ненулевого бита кода изображения и, соответственно, значения счетчиков, которые автомат подает двум группам автоматов: первое — группе из n, второе — группе из m автоматов. Затем ждет пока не обнулятся счетчики всех автоматов в двух цепочках и переходит в следующее состояние. Через несколько тактов после того, как автомат пройдет по всем своим состояниям, на экране возникнет черно-белая конфи-

гурация. Если теперь подавать на свободные входы нулевые векторы, то можно сохранять эту конфигурацию сколь угодно долго. То есть это изображение. Так как каждая нарисованная точка соответствует точке из поданного на вход кода, то полученное изображение соответствует данному двоичному коду. Лемма доказана.

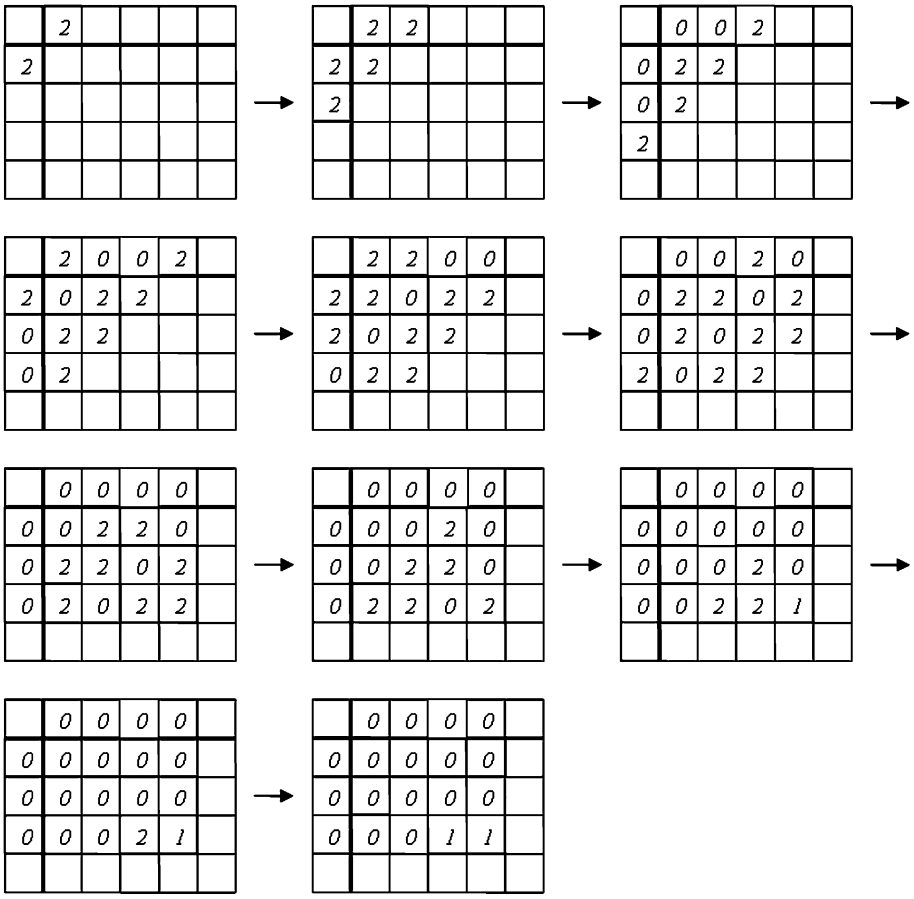

Рис. 6. Пример конструирования двух точек.

Из лемм 1 и 2 следует утверждение **Теоремы 1**.

На рисунке 6 приведен пример построения точек с координатами

$(3,3)$ и $(3,4)$. Для построения точки мы посылаем волну (две диагонали) из двоек. Волна продвигается на одну диагональ за такт. Если послать две волны с интервалом в одну пустую диагональ, то так как они движутся с одинаковой скоростью, одна не сможет догнать другую. Диагональ исчезает за 2 такта: остается одна двойка, которая в следующий момент становится единицей. То есть одна волна успеет исчезнуть до того, как вторая догонит ее, и от нее не останется двоек, которые могли бы повлиять на формирование очередной точки. Значит, на каждый третий такт мы можем начинать посылать сигнал для следующей точки. Будем называть этот алгоритм — *Алгоритм 1.*

Оценим время работы *Алгоритма 1.* На точку с координатами (i,j) автомат тратит $i+j+1$ такт. Изображение может содержать не более чем mn точек. Последнюю из них автомат начнет конструировать не позже чем через $3mn$ тактов после начала подачи последовательностей для первой точки (так как через каждые 3 такта автомат начинает конструировать следующую точку). Плюс один такт на то, чтоб последняя точка успела стать единицей. Если точек меньше чем nm, например k, то последнюю из них автомат начнет конструировать через $3(k-1)$ тактов после начала подачи последовательностей для первой точки. И тогда последняя точка успеет достроиться за счет разницы между $3mn$ и $3k$. Следовательно,

$$T(n,m,3) \leqslant 3mn + 1.$$

Также можем оценить время построения по *Алгоритму 1* изображения, содержащего $k \leqslant mn$ точек. Будем обозначать его через $T_k(n,m,3)$. Мы посчитаем общее время подачи последовательностей для построения точек, то есть $3k$. Также необходимо добавить время на очистку экрана, на это достаточно 4 такта. Осталось добавить время на то, чтобы последняя точка дорисовалась, когда автомат уже стал подавать только нулевые последовательности. На это необходимо $i+j+1$ такт, где (i,j) — координаты последней точки. Оценим $i+j+1 \leqslant n+m+1$ и получим

$$T_k(n,m,3) \leqslant \min(3k+n+m+5, 3nm+1).$$

3. Точное значение времени построения изображения при неограниченном числе состояний клеточных автоматов

Лемма 3. *Если* $m \geqslant n \geqslant 3$, $n, m \in \mathbb{N}$, *то* $T(n, m) \geqslant n$.

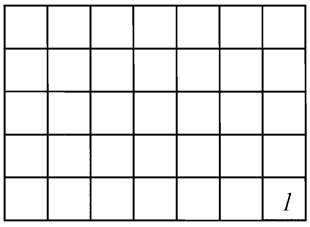

Рис. 7. Пример изображения, которое невозможно сконструировать быстрее чем за n тактов.

Доказательство. Приведем пример изображения, которое нельзя сконструировать быстрее чем за n тактов. Это изображение, состоящее из одной единственной точки с координатами (n, m) (см. рис. 7). Допустим, что мы уже нарисовали эту точку. Это означает, что был такой момент, когда не все входы у нее были нулевыми. А значит, еще раньше был момент, когда не все входы у автоматов из ее окрестности были нулевыми и так далее. Так как каждый автомат видит только состояния соседних с ним автоматов, то за один такт ненулевое состояние распространяется не более чем на одну клетку в каждом направлении. Это означает, что клетки с координатами (n, m) оно достигнет не раньше чем через n тактов. Лемма доказана.

Лемма 4. *Если* $m \geqslant n \geqslant 3$, $n, m \in \mathbb{N}$, *то*

$$T(n, m) \leqslant n.$$

Доказательство. Построим генератор, который конструирует любое изображение за время n. Опишем клеточный автомат. Его состояние состоит из двух компонент: первая — счетчик (от 0 до $n - 1$), а

вторая — метка (0 или 1). В начальный момент времени счетчик автомата и метка равны нулю. Существенным для клеточного автомата является только состояние верхнего автомата. Если значение счетчика автомата равно нулю, автомат выдает метку. В каждый момент времени автомат берет значение счетчика и метку у верхнего автомата. Если значение счетчика больше нуля, автомат уменьшает его на 1, присваивает это значение своему счетчику и берет новое значение метки. Таким образом определяется новое состояние клеточного автомата. Если значение счетчика верхнего автомата равно 0, автомат не меняет значение своего счетчика и своей метки. Таким образом, клеточные автоматы имеют по $2n$ состояний, $E_{2n} = \{0, 1, \ldots, 2n-1\}$. Для состояний 0 и 1 значения счетчика равны 0, значения метки равны 0 и 1 соответственно. Для состояний с 2-го по n-е значение метки равно 0 и значения счетчика равно $1, 2, \ldots, n-1$ соответственно. Для состояний с $(n+1)$-го по $(2n-1)$-е значение метки равно 1 и значения счетчика равно $1, 2, \ldots, n-1$ соответственно. Переходы состояний описаны выше.

Опишем внешний автомат соответствующий двоичному коду K. В начальном (нулевом) состоянии автомат подает на верхнюю горизонталь n-ю строку кода изображения и значение счетчика n. В следующий такт — $(n-1)$-ю строку изображения и значение счетчика $(n-1)$ и так далее, в n-й такт подает первую строку и значение счетчика 1, в $(n+1)$-й такт возвращается в начальное состояние. То есть в i-м состоянии автомат подает $(n-i)$-ю строку и значение счетчика $(n-i)$. Таким образом, внешний автомат имеет n состояний.

Поясним этот алгоритм. Внешний автомат подает последовательности из кода изображения с соответствующими счетчиками с верхней границы в соответствующие столбцы экрана (на левую границу всегда подаются нули). Сначала подает метку самой нижней клетки столбца и значение счетчика n, затем метку $(n-1)$-й клетки столбца и значение счетчика $(n-1)$ и т. д. За один такт сигнал продвигается на одну клетку вниз и значение счетчика уменьшается на 1. Поэтому, когда сигнал достигнет нижней строки, счетчик обнулится и автомат выдаст нужную метку. Аналогично со всеми остальными клетками столбца. Так как значения счетчиков подавались каждый раз на еди-

ницу меньше, то все счетчики обнулятся одновременно на n-м такте. То есть на n-м такте на экране построится изображение, соответствующее заданному двоичному коду. Значит, $T(n, m) \leqslant n$. Лемма доказана.

Из последних двух лемм следует утверждение **Теоремы 2**.

4. Верхняя оценка времени построения изображения при ограниченном числе состояний клеточных автоматов

В этом разделе мы приведем алгоритм, который при ограниченном числе состояний клеточных автоматов работает в константу раз быстрее чем *Алгоритм 1.*

Теорема 4. *Если* $m \geqslant n \geqslant 3$, $n, m, k \in \mathbb{N}$, *то*

$$T(n, m, 2^{k+1} - 1) \leqslant (m - k + 2) \cdot]n/k[+ k - 1.$$

Доказательство. *Алгоритм 2.*

Опишем клеточный автомат. Состояния 0 и 1 — метки, 2 — вспомогательное состояние. Остальные состояния являются кодами изображения, стоящего в столбце из k клеток, $(k-1)$ клетки, ..., 3 и 2 клеток соответственно. Объединим эти состояния в группы с номером, соответствующим количеству клеток, информацию о которых содержат эти состояния. Состояния закодируем двоичным кодом длины j, $1 < j \leqslant k$. Такому состоянию соответствует столбик высоты j с соответствующим двоичным кодом изображения в нем (код читается снизу вверх). Кроме того такому состоянию соответствует состояние с кодом длины $(j-1)$, которое получается из исходного отбрасыванием старшего разряда, и метка, равная значению старшего разряда. Таким ообразом, количество состояний равно $1 + 2 + 2^2 + 2^3 + \ldots + 2^{k-1} + 2^k = 2^{k+1} - 1$.

Мы будем подавать сверху на горизонталь соответствующие номера состояний, а из них клеточные автоматы восстановят целый блок изображения размером $k \times m$ клеток.

Опишем функцию переходов состояний.

Состояние 0. Остается неизменным, если на всех входах стоят только нули и единицы. Если у автомата верхний вход u из k-й группы, остальные входы нулевые или единичные, то автомат переходит в состояние u. Если у автомата нижний вход d из j-й группы, $j \in \{(k-1),(k-2),\ldots,2$, остальные входы нулевые, то автомат переходит в соответствующее состоянию d состояние d' из $(j-1)$-й группы. Если у автомата левый вход равен 2, остальные входы нулевые или единичные, то автомат переходит в состояние 2. В остальных случаях состояние 0 остается неизменным.

Состояние 1 изменяется так же, как состояние 0, то есть остается неизменным, если на всех входах стоят только нули и единицы. Если у автомата нижний вход d из j-й группы, $j \in \{(k-1),(k-2),\ldots,2$, остальные входы нулевые, то автомат переходит в соответствующее состоянию d состояние d'. Если у автомата левый вход равен 2, остальные входы нулевые или единичные, то автомат переходит в состояние 2. В остальных случаях состояние 1 остается неизменным.

Состояние 2. Если у автомата верхний вход u из k-й группы, то автомат переходит в состояние 0 или 1, соответствующее коду u. В остальных случаях остается неизменным.

Состояния k-й группы. Если у автомата на нижнем входе стоит 2, то автомат переходит в соответствующее состояние из $(k-1)$-й группы. Иначе автомат переходит в нулевое состояние.

Состояния j-й группы, $j \in \{(k-1),(k-2),\ldots,2\}$ всегда переходит в состояние 0 или 1, соответствующее коду своего состояния в настоящий момент времени.

Опишем внешний автомат, соответствующий двоичному коду K. Для нижних k горизонталей он составляет m-мерный вектор V_m состояний из k-й группы (i-й элемент вектора — бинарный код изображения из i-го столбца, $1 \leqslant i \leqslant m$). С первого по m-й такты автомат подает с левой границы экрана n-мерные векторы, в которых на n-м месте стоит двойка, на остальных местах — нули. С верхней границы экрана автомат подает нулевые векторы во все такты кроме $(m-(n-1))$-го — в этот момент он подает вектор V_m. Через $k-1$ такт после того, как сигналы встретятся с горизонталью из двоек,

все состояния расшифруются и в этом блоке восстановится нужное изображение. Следующая горизонталь из двоек — с номером $n-k$. Ее можно начинать подавать как только вектор V_m пройдет ниже ее, то есть с $(m-(k-2))$-го такта. То есть вектор подаваемый с левой границы экрана с $(m-(k-2))$-го по m-й такты содержит две двойки: на m-м и $n-k$-м местах. И так далее мы построим все изображение.

Время работы описанного алгоритма:

$$T(n,m,2^{k+1}-1) \leqslant (m-(k-2))\cdot]n/k[+(k-2)+(k-1)$$

или

$$T(n,m,2^{k+1}-1) \leqslant (m-k+2)\cdot]n/k[+2k-3.$$

Теорема доказана.

В заключение автор выражает благодарность Э. Э. Гасанову за постановку задачи и научное руководство.

Список литературы

[1] Кудрявцев В. Б., Подколзин А. С., Болотов А. А. Основы теории однородных структур. М.: Наука, 1990.

[2] Кудрявцев В. Б., Алешин С. В., Подколзин А. С. Введение в теорию автоматов. М.: Наука, 1985.