ИПМ им. М.В. Келдыша РАН • Электронная библиотека
Математические вопросы кибернетики • Выпуск 21

Э. Э. Гасанов

Клеточные автоматы с локаторами как модель устройств с беспроводной связью

Рекомендуемая форма библиографической ссылки: Гасанов Э. Э. Клеточные автоматы с локаторами как модель устройств с беспроводной связью // Математические вопросы кибернетики. Вып. 21. — М.: ФИЗМАТЛИТ, 2023. — С. 5–51. URL: https://library.keldysh.ru/mvk.asp?id=2023-5
DOI: 10.20948/mvk-2023-5

КЛЕТОЧНЫЕ АВТОМАТЫ С ЛОКАТОРАМИ КАК МОДЕЛЬ УСТРОЙСТВ С БЕСПРОВОДНОЙ СВЯЗЬЮ

Э. Э. ГАСАНОВ

(МОСКВА)

§ 1. Развитие автоматной модели

Автомат — это устройство, на вход которого поступает некоторая последовательность символов из некоторого входного алфавита, а на выходе получается, вообще говоря, другая последовательность символов из некоторого, возможно, другого выходного алфавита. Автомат функционирует в дискретном времени, т. е. считается, что i-й элемент входной последовательности поступает в i-й момент времени. В эти же моменты на выходе автомата появляются выходные символы. Автомат обладает свойством детерминированности, т. е. значение выходного символа в i-й момент времени зависит только от входных символов до i-го момента времени включительно и не зависит от входных символов, которые поступят в последующие моменты времени. Входные и выходные последовательности бывают как конечные, так и бесконечные. В первом случае эти последовательности называют входными и выходными словами, а во втором — сверхсловами. При этом если на вход автомата поступило слово (конечное) длины n, то на выходе обязательно получается слово той же длины n. Если выходной символ в i-й момент времени однозначно определяется по входному символу в i-й момент времени, то говорят об автоматах без памяти, в противном случае говорят об автоматах с памятью. Явление запоминания в автомате реализуется через понятие состояния автомата, т. е. считается, что автомат под влиянием входных воздействий переходит в некоторое состояние, а выходной символ зависит от состояния автомата и входного символа в текущий момент времени. Если у автомата конечное число состояний, то говорят о конечном автомате, в противном случае — о бесконечном.

Поскольку любое цифровое устройство, любой чип и любая компьютерная программа являются автоматами, то понятно, что теория автоматов служит основой всего цифрового мира.

Перейдем к формальному определению понятия автомата.

Пусть A — некоторое конечное множество, называемое *алфавитом*. Элементы множества A будем называть *буквами* или *символами*.

Словом в алфавите A будем называть конечную последовательность элементов алфавита A. Количество элементов в последовательности будем называть *длиной слова*. Длину слова α будем обозначать $|\alpha|$.

DOI: 10.20948/mvk-2023-5

Слово длины 0 будем называть *пустым* и обозначать Λ.

Бесконечные последовательности букв алфавита A будем называть *сверхсловами* в алфавите A.

Если α — слово или сверхслово, n — натуральное число, не превышающее длины слова α, то n-ю букву слова или сверхслова α будем обозначать $\alpha(n)$.

Через $pref(\alpha, n)$ будем обозначать *префикс* длины n слова или сверхслова α, т. е. $pref(\alpha, n) = \alpha(1)\ldots\alpha(n)$, для префикса также будет использоваться обозначение $\alpha]_n$, где n — натуральное число, не превышающее длины слова α.

Обозначим через $\alpha\beta$ конкатенацию слова α и слова или сверхслова β, т. е. слово или сверхслово, полученное присоединением к последовательности α последовательности β справа.

Обозначим $a^n = \underbrace{a\ldots a}_{n}$, здесь a может быть буквой алфавита A или словом в алфавите A, n — натуральное. Обозначим a^∞ сверхслово, состоящее из периодически повторяемого слова a.

Если α и β — слова, то запись $\alpha\beta^\infty$ будет обозначать сверхслово, в начале которого идет слово α, а затем слово β периодически повторяется бесконечное число раз. При этом слово α будем называть *предпериодом* сверхслова $\alpha\beta^\infty$, а слово β — *периодом* сверхслова $\alpha\beta^\infty$.

Через A^* будем обозначать множество всех слов в алфавите A. По определению будем считать, что пустое слово Λ принадлежит A^*. Через A^∞ будем обозначать множество всех сверхслов в алфавите A.

В формальном определении автомата участвуют 3 алфавита: A — *алфавит входных символов*, B — *алфавит выходных символов* и Q — *алфавит состояний*.

Автомат определяет функцию, которая слова или сверхслова в алфавите A переводит в слова или сверхслова в алфавите B, причем эта функция детерминированная и слово длины n переводит слово той же длины n. Как мы уже отмечали, выходной символ автомата в момент времени t зависит от состояния автомата и входного символа в момент времени t, т. е. у автомата имеется функция $\psi : Q \times A \to B$, называемая *функцией выходов*. Также у автомата имеется *функция переходов* $\varphi : Q \times A \to Q$, которая по текущим состоянию и входному символу определяет состояние в следующий момент времени.

Следуя [15, 16], определим, что *конечный автомат* — это пятерка:

$$V = (A, Q, B, \varphi, \psi),$$

где A — входной конечный алфавит, Q — конечное множество состояний, B — выходной конечный алфавит, $\varphi : Q \times A \to Q$ — функция переходов, $\psi : Q \times A \to B$ — функция выходов.

Везде далее мы будем изучать только конечные автоматы, поэтому часто будем использовать термин «автомат» вместо термина «конечный автомат».

$V_{q_0} = (A, Q, B, \varphi, \psi, q_0)$ — *инициальный автомат* с начальным состоянием q_0. Будем также писать $V(q_0)$ для обозначения инициального автомата V_{q_0}.

Если на вход инициальному автомату V_{q_0} подается слово или сверхслово x, на выходе получается слово или сверхслово y и $q(t)$ означает состояние автомата в момент времени t, то функционирование автомата задается системой

$$\begin{cases} q(1) = q_0, \\ q(t+1) = \varphi\big(q(t), x(t)\big), \\ y(t) = \psi\big(q(t), x(t)\big), \end{cases}$$

где $t = 1, 2, \ldots$. Такая система называется *системой канонических уравнений* автомата.

Расширим функции φ и ψ на $Q \times A^*$, а именно, если $\alpha \in A^*$, $a \in A$, то индуктивно определим

$$\varphi(q, \alpha a) = \varphi(\varphi(q, \alpha), a),$$

$$\psi(q, \alpha a) = \psi(\varphi(q, \alpha), a).$$

Введем также обозначения

$$\overline{\varphi}(q, \alpha) = \varphi(q, \alpha]_1)\varphi(q, \alpha]_2) \ldots \varphi(q, \alpha),$$

$$\overline{\psi}(q, \alpha) = \psi(q, \alpha]_1)\psi(q, \alpha]_2) \ldots \psi(q, \alpha),$$

если α — слово, если же α — сверхслово, то

$$\overline{\varphi}(q, \alpha) = \varphi(q, \alpha]_1)\varphi(q, \alpha]_2) \ldots \varphi(q, \alpha]_n) \ldots,$$

$$\overline{\psi}(q, \alpha) = \psi(q, \alpha]_1)\psi(q, \alpha]_2) \ldots \psi(q, \alpha]_n) \ldots.$$

Функция $\overline{\psi}(q_0, \alpha)$ — это и есть словарная функция, реализуемая инициальным автоматом V_{q_0}, которая слова или сверхслова в алфавите A переводит в слова или сверхслова в алфавите B. Такого рода функции называются *автоматными функциями*.

В теории автоматов особую роль играет автомат, называемый *задержкой* или *задержкой с нулевым начальным состоянием* и изображаемый, как приведено на рис. 1.

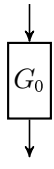

Рис. 1. Изображение автомата «задержка»

Задержка — это автомат, который в первый момент выдает на выходе число 0, а в следующие моменты выдает на выходе числа, которые он получил на входе в предыдущие такты. Формально задержка задается следующей системой канонических уравнений:

$$\begin{cases} q(1) = 0, \\ q(t+1) = x(t), \\ y(t) = q(t), \end{cases}$$

где $t = 1, 2, 3, \ldots$, $x(t), q(t), y(t) \in \{0, 1\}$.

Задержка является простейшим элементом для запоминания. В частности, именно на задержках можно сохранять состояние автомата.

Одним из способов описания автоматов является представление автоматов в виде схем из функциональных элементов и задержек, где функциональные элементы реализуют некоторые булевы функции. Такие схемы из функциональных элементов и задержек называются *автоматными схемами*, или *структурными автоматами*. Если в базисе нет элемента задержки, а есть только функциональные элементы, то такие схемы называются *схемами из функциональных элементов* (СФЭ). СФЭ реализуют булевы операторы.

Первым этапом синтеза интегральных схем является синтез структурного автомата, реализующего необходимую автоматную функцию.

В работе [21] О. Б. Лупанов показал, что сложность реализации СФЭ в стандартном базисе для почти всех булевых функций от n переменных асимптотически равна $2^n/n$. Здесь под сложностью СФЭ подразумевается число функциональных элементов в схеме. В статье [1] М. Н. Вайнцвайг ввел еще одну меру сложности — мощность СФЭ. Согласно определению Вайнцвайга *мощность* СФЭ равна максимальному количеству элементов схемы, выдающих на выходе 1, где максимум берется по всем входным наборам. Вайнцвайг показал, что для стандартного базиса мощность СФЭ для булевых функций от n переменных в худшем случае равна по порядку n. В работе Г. В. Калачева [12] мощность определяется как среднее количество элементов схемы, выдающих на выходе 1, где среднее берется по всем входным наборам. Еще одной важной мерой сложности СФЭ является ее *глубина*, которая равна максимальному числу элементов, встречающихся на пути, ведущем от входа схемы к ее выходу. Если сложность моделирует размер устройства, а мощность — ее энергопотребление, то глубина определяет тактовую частоту устройства. С. Б. Гашков [8] и С. А. Ложкин [20] показали, что функция Шеннона глубины СФЭ для функций от n переменных равна $n-\log_2\log_2 n$ с точностью до аддитивной константы.

Задачу синтеза структурного автомата легко свести к задаче синтеза СФЭ, реализующей некоторый булев оператор.

В самом деле, пусть $f(x_1,\ldots,x_n)$ — произвольная автоматная функция, заданная канонической системой

$$\begin{cases} q_1(1)=0,\ldots,q_r(1)=0,\\ q_i(t+1)=\varphi_i(q_1(t),\ldots,q_r(t),x_1(t),\ldots,x_n(t)),\ i=1,\ldots,r,\\ y(t)=\psi(q_1(t),\ldots,q_r(t),x_1(t),\ldots,x_n(t)). \end{cases}$$

Пусть схема S реализует систему булевых функций $\varphi_1,\ldots,\varphi_r$, ψ, зависящих от переменных $q_1,\ldots,q_r,x_1,\ldots,x_n$. Тогда нетрудно видеть, что автоматная функция $f(x_1,\ldots,x_n)$ может быть представлена схемой, изображенной на рис. 2.

Тем самым метод синтеза Лупанова позволяет строить «хорошие» с точки зрения количества элементов структурные автоматы.

Вторым этапом синтеза интегральных схем является укладка функциональных элементов и задержек на плоскость и разводка проводов. Эвристические алгоритмы, используемые для укладки элементов и разводки проводов, показывают, что провода занимают существенную площадь и что эффект, полученный от асимптотически оптимального алгоритма Лупанова,

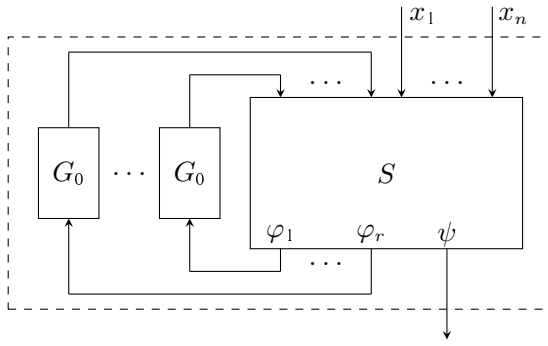

Рис. 2. Реализация автоматной функции $f(x_1, \ldots, x_n)$

нивелируется. Возникает вопрос: может быть, мы просто не умеем хорошо укладывать элементы и разводить провода и существует метод, который позволит избежать существенного увеличения площади из-за проводов?

Ответ на этот вопрос дает модель плоских клеточных схем, предложенных С. С. Кравцовым [14]. Фактически плоская схема является укладкой СФЭ на целочисленную решетку на плоскости таким образом, что соединенными могут быть только элементы, находящиеся в соседних клетках. При этом длинные провода моделируются цепью элементов, реализующих тождественные функции. Кравцов доказал, что площадь плоских схем, реализующих булевы функции от n переменных, равна по порядку 2^n. Откуда с учетом результата Лупанова следует, что основную площадь занимают коммутационные элементы, т. е. провода. Г. В. Калачев в статье [12] показал, что произвольную булеву функцию от n переменных можно реализовать плоской клеточной схемой с площадью порядка 2^n, глубиной порядка n и средней мощностью порядка $2^{n/2}$. При этом указанные параметры оптимальны по порядку для почти всех булевых функций.

Все равно остаются сомнения: ведь плоские схемы Кравцова не отражают реалии современных технологий создания интегральных схем. В реальных интегральных схемах функциональные элементы лежат на одном слое, а провода на других слоях, причем над функциональным элементом может проходить много проводов. Может быть, эти технологии позволят принципиально сократить площадь, занимаемую проводами? Ответ на этот вопрос опять отрицательный, и он дается в работе Т. Р. Сытдыкова [25]. В этой работе вводятся многослойные плоские схемы, в которых функциональные элементы лежат на первом слое, а провода на остальных. В этой статье Сытдыков доказал, что сложность реализации почти всех булевых функций от n переменных k-слойными схемами равна по порядку $\frac{2^n}{\min(n, 2\log_2 k)}$. Тем самым при фиксированном числе слоев порядок площади схем сохраняется и принципиального выигрыша за счет многослойности получить нельзя.

Рассмотренные выше клеточные схемы не содержали элемента «задержка» и поэтому реализовывали булевы функции и операторы, тогда как, чтобы реализовывать автоматные функции, нам обязательно надо будет использовать задержки. Хоть выше мы и показали, что задачу реализации автоматной функции можно свести к задаче реализации булевого оператора,

но на самом деле это не совсем так, поскольку часть входов этого оператора будут выходами задержек и, значит, не будут независимыми. Может быть, использование этой зависимости поможет сократить площадь схем? Пока глобального результата такого рода нет, но есть частный результат, который говорит: использование знания, что мы реализуем автоматную функцию, может привести к существенному снижению средней мощности схем.

Например, если рассмотреть задачу реализации автономных автоматных функций с 2^n состояниями, то стандартный подход, представленный рис. 2, даст площадь плоской схемы, равную по порядку 2^n, а мощность — $2^{n/2}$. Тогда как в статье [5] А. С. Воротников показал, что автономный автомат с 2^n состояниями можно реализовать плоской автоматной схемой с площадью, равной по порядку 2^n, и средней мощностью, равной по порядку $\frac{2^{n/2}}{n}$, т. е. среднюю мощность можно сократить в n раз, не увеличивая площадь.

§ 2. Клеточные автоматы

Математической моделью, где нет проблемы разводки проводов, является модель клеточного автомата. Клеточный автомат — это тоже клеточная схема, в клетках которой находятся одинаковые автоматы, и провода соединяют единообразно каждый автомат с некоторыми его соседями, что автоматически снимает проблему разводки проводов, поскольку в этом случае она будет носить локальный характер.

Понятие клеточного автомата (другие названия — самовоспроизводящиеся автоматы и однородные структуры) возникло в результате усовершенствования модели Дж. фон Неймана [24, 32, 33], предложенной им для описания процессов самовоспроизведения в биологии и технике, и в описанном ниже виде использовалось А. Берксом [28], Э. Муром [23], В. Б. Кудрявцевым, А. С. Подколзиным, А. А. Болотовым [17] и другими исследователями.

Клеточный автомат — это математический объект с дискретными пространством и временем. Каждое положение в пространстве представлено отдельной клеткой, а каждый момент времени — дискретным временным шагом или поколением. Состояние каждой пространственной клетки определяется очень простыми правилами взаимодействия. Эти правила предписывают изменения состояния каждой клетки в следующем такте времени в ответ на текущее состояние соседних клеток. При этом для разных клеток правила изменения состояний могут быть разными.

Если в качестве преобразователя информации, стоящего в клетке пространства, выбрать конечный автомат, причем один и тот же для всех клеток, то мы приходим к понятию однородной структуры. В таком случае содержательно клеточный автомат представляет собой бесконечную автоматную схему, построенную следующим образом. Рассмотрим k-мерное евклидово пространство. Разобьем его на гиперкубы с единичным ребром, ребра которых параллельны осям координат. В каждый гиперкуб поместим один и тот же конечный автомат V с m входами и одним выходом. Разветвим выход автомата и соединим с входами его соседей одинаковым образом для всех гиперкубов в пространстве. Получим бесконечную, однородным

образом устроенную автоматную схему, которая и называется клеточным автоматом. Последовательность состояний отдельных автоматов V, содержащую состояния всех автоматов схемы, будет образовывать состояние клеточного автомата. Последовательность состояний клеточного автомата, возникающая при синхронной работе всех составляющих его конечных автоматов, называется функционированием клеточного автомата.

Клеточные автоматы представляют собой дискретную математическую модель широкого класса реальных систем вместе с протекающими в них процессами, таких как физические среды, в которых реализуются тепловые и волновые явления, химические растворы с реакциями в них, биологические ткани, в которых происходит обмен веществ, технические схемы управления, производящие переработку механических и электрических сигналов, вычислительные схемы и т.п.

Если задать начальные состояния автоматов, то в схеме начнется изменение состояний автоматов, определяемое законами функционирования автоматов и связями между ними. Явление глобального изменения этих состояний и является главным объектом изучения в теории клеточных автоматов.

Дадим формальное определение понятия «клеточный автомат», следуя [16, 17].

Клеточным автоматом называется четверка $\sigma = (\mathbb{Z}^k, Q, V, \varphi)$, где $\mathbb{Z}^k$ — множество k-мерных векторов с целыми координатами; Q — некоторое множество состояний, в котором выделено одно состояние q_0, называемое *состоянием покоя*; $V = (\alpha_1, \ldots, \alpha_{h-1})$ — упорядоченный набор попарно различных ненулевых векторов из $\mathbb{Z}^k$; φ — функция, зависящая от переменных $x_0, x_1, \ldots, x_{h-1}$, $\varphi : Q^h \to Q$, $\varphi(q_0, \ldots, q_0) = q_0$. Элементы множества $\mathbb{Z}^k$ называются *ячейками* клеточного автомата σ; элементы множества Q называются *состояниями ячейки* клеточного автомата σ; набор V называется *шаблоном соседства* клеточного автомата σ; наконец, функция φ называется *локальной функцией переходов* клеточного автомата σ. Условие $\varphi(q_0, \ldots, q_0) = q_0$ интерпретируется как условие сохранения состояния покоя.

Здесь нам нужно было вводить упорядочение шаблона соседства V для того, чтобы установить взаимно-однозначное соответствие между векторами из V и переменными локальной функции переходов φ. Это соответствие можно сделать более явным, если индексировать переменные функции φ самими векторами, т. е. считать, что локальная функция переходов φ зависит от переменных $x_0, x_{\alpha_1}, \ldots, x_{\alpha_{h-1}}$, здесь индекс первой переменной есть нулевой вектор $0 = (0, \ldots, 0) \in \mathbb{Z}^k$. Если договориться так индексировать переменные локальной функции переходов, то их можно записывать в любом порядке, и тогда можно воспринимать шаблон соседства просто как множество, а не упорядоченный набор.

В дальнейшем мы так и будем поступать: воспринимать шаблон соседства как множество векторов и индексировать переменные локальной функции переходов векторами из шаблона соседства, при этом в индексах внешние круглые скобки у векторов часто будем опускать. Например, если $k = 2$, $Q = \{0, 1\}$ и $V = \{(-1, 0), (0, -1), (1, 0), (0, 1)\}$, то пример локальной функции переходов может выглядеть так: $\varphi = (x_{-1,0} \vee x_{0,-1}) \oplus x_{0,0} \oplus x_{1,0} \oplus x_{0,1}$. Здесь символом «$\oplus$» обозначено сложение по модулю 2.

Если $\alpha \in \mathbb{Z}^k$ — ячейка клеточного автомата σ, то множество $V(\alpha) = \{\alpha, \alpha+\alpha_1, \ldots, \alpha+\alpha_{h-1}\}$ называется *окрестностью ячейки* α.

Состоянием клеточного автомата σ называем произвольную функцию f, определенную на множестве $\mathbb{Z}^k$ и принимающую значения из Q. Такую функцию можно интерпретировать как некую мозаику, возникающую в k-мерном пространстве в результате приписывания каждой точке с целочисленными координатами некоторого состояния из множества Q. Множество всевозможных состояний клеточного автомата обозначим Σ.

Если $\alpha \in \mathbb{Z}^k$, f — состояние клеточного автомата σ, то значение $f(\alpha)$ называем *состоянием ячейки* α, *определяемым состоянием* f *клеточного автомата* σ.

На множестве Σ определим *глобальную функцию переходов* Φ клеточного автомата σ, полагая $\Phi(f) = g$, где $f, g \in \Sigma$ и для любой ячейки $\alpha \in \mathbb{Z}^k$ выполняется тождество

$$g(\alpha) = \varphi(f(\alpha), f(\alpha+\alpha_1), \ldots, f(\alpha+\alpha_{h-1})). \tag{1}$$

Содержательная интерпретация отображения Φ такова, что состояние каждой ячейки «после перехода» определяется по состоянию упорядоченной окрестности ячейки «до перехода» с помощью локального закона φ одинаково для всех ячеек.

Поведениями клеточного автомата σ называем такие последовательности $f_0, f_1, f_2, \ldots$ ее состояний, для которых выполняется $f_{i+1} = \Phi(f_i)$ для всех $i = 0, 1, 2, \ldots$, причем f_i называется *состоянием клеточного автомата* σ *в момент* i, а f_0 также называется *начальным состоянием клеточного автомата* σ.

Состояние клеточного автомата, у которого лишь конечное число ячеек находится в состоянии, отличном от состояния покоя, назовем *конфигурацией*. Множество конфигураций будем обозначать через Σ'.

Если задано некоторое состояние клеточного автомата, то ячейки, находящиеся в состоянии, отличном от состояния покоя, будем называть *активными*.

Клеточные автоматы являются мощнейшими вычислителями, потому что состоят из огромного числа независимых элементарных вычислителей, которые могут работать параллельно. Удобным с точки зрения технической реализации является то, что все связи имеют локальный характер, т. е. провода проводятся только между соседними клетками (клетками из шаблона соседства), и тем самым в клеточных автоматах нет проблемы разводки проводов.

На основе модели клеточных автоматов были построены такие технические устройства, как ПЛИС (программируемая логическая интегральная схема), или в англоязычной версии FPGA (field-programmable gate array).

Клеточные автоматы идеально подходят для реализации разностных схем, и пример такой реализации можно найти в [16, с. 170]. Также клеточные автоматы удобны при реализации конвейерных вычислений и моделировании процессов в пространстве, например аэродинамических процессов (см. статью А. С. Гордеевой [9]). Но наличие только локальных проводов и отсутствие длинных проводов приводит к тому, что время распространения сигнала от одной клетки к другой всегда пропорционально расстоянию

между клетками. Это приводит к тому, что процессами в клеточных автоматах трудно управлять, трудно синхронизировать действия отдельных элементарных автоматов.

Эту проблему хорошо можно продемонстрировать на примере задачи о синхронизации стрелков.

Задача синхронизации стрелков впервые предложена Дж. Майхиллом в 1957 году и опубликована (с решением) в 1968 году Ф. Муром [31].

В этой задаче рассматривается одномерный клеточный автомат на $\mathbb{Z}^1$. Шаблон соседства равен $V = \{-1, 1\}$, т. е. каждая ячейка имеет двух соседей слева и справа. Множество состояний имеет как минимум три состояния: 0 — состояние покоя, 1 — солдат в начальном состоянии, 2 — огонь. Имеется ограничение на функцию переходов, что солдат в начальном состоянии, имея соседями таких же солдат, состояния не меняет, т. е. $\varphi(1, 1, 1) = 1$. Начальная конфигурация представляет собой непрерывный отрезок из r ячеек в состоянии 1 (солдаты), а все остальные ячейки находятся в состоянии покоя 0. Надо, чтобы в какой-то момент все активные ячейки перешли в состояние 2 одновременно (выстрелили).

Стандартное решение этой задачи описывает две волны состояний, распространяющиеся по ряду солдат, одна из которых движется в три раза быстрее другой. Быстрая волна отражается от дальнего края ряда и встречается с медленной в центре. После этого две волны разделяются на четыре, движущиеся в разные стороны от центра. Процесс продолжается, каждый раз удваивая число волн, пока длина отрезков ряда не станет равной 1. В этот момент все солдаты стреляют. Это решение требует $3r$ единиц времени для r солдат.

Получается, чтобы заставить колонию элементарных автоматов что-то сделать одновременно, нам нужно несколько раз по колонии пробежаться.

Приведем еще один пример задачи, которая в клеточных автоматах решается просто. Это **задача нахождения выпуклой оболочки.** Постановка задачи: изначально на плоскости ячейки, принадлежащие некой связной фигуре, окрашены в черный цвет (находятся в некотором специальном состоянии), ячейки в состоянии покоя считаются окрашенными белым цветом; нужно окрасить в серый цвет минимальный прямоугольник, включающий заданную фигуру. Задача может быть решена с помощью клеточного автомата с шаблоном соседства «крест» ($V = \{(-1, 0), (1, 0), (0, -1), (0, 1)\}$) и функцией переходов, в которой белая ячейка клетка окрашивается в серый цвет, если у нее есть по крайней мере 2 черных или серых соседа. Время, за которое будет построен охватывающий прямоугольник, в худшем случае приблизительно равно полупериметру этого прямоугольника. Худший случай будет достигаться, например, на начальной конфигурации, состоящей из двух перпендикулярных отрезков длины n, образующих угол. В этом случае охватывающим прямоугольником будет квадрат со стороной n, и он будет построен за время $2n - 3$. Поскольку конечная конфигурация состоит из n^2 активных ячеек, то в среднем за такт будет зажигаться порядка n ячеек, т. е. только малая часть вычислителей действительно будет работать.

Заметим, что в данном решении ни одна из ячеек не «понимает», когда процесс построения выпуклой оболочки завершен. А бывают случаи, когда такое «понимание» необходимо, например в задаче назначения командира.

Задача назначения командира. Постановка задачи: начальная конфигурация — связная фигура из черных клеток; финальная конфигурация

должна быть такой, чтобы любая одна черная клетка стала красной, и эту красную клетку называют командиром. В работе [22] М. Ф. Музаффарова показала, что эту задачу можно решить за время, пропорциональное полупериметру охватывающего прямоугольника. Идея решения заключается в том, чтобы построить выпуклую оболочку серого цвета (клетки основной фигуры остаются черными). Когда выпуклая оболочка построена, проходимся по верхнему краю выпуклой оболочки и выбираем самую левую верхнюю черную клетку в качестве командира. Значит, верхняя левая точка выпуклой оболочки должна «понять», что она является верхней левой точкой и что выпуклая оболочка уже построена.

Чтобы понять, что выпуклая оболочка построена, предлагается запустить из верхней левой точки огибающий сигнал. Огибающий сигнал — это совокупность из 8 состояний, по 2 на каждую сторону прямоугольника, один из которых для запоминания, что он был раньше черной клеткой, а второй — для серой клетки. Он сначала бежит, оставляя след, вправо по верхней стороне, затем вниз по правой стороне, затем влево по нижней стороне и, наконец, вверх по левой стороне. Момент, когда он вернется в верхнюю левую точку, и будет означать, что выпуклая оболочка построена. Беда в том, что каждая ячейка, у которой слева и сверху находятся ячейки в состоянии покоя, будет считать себя верхней левой точкой. Например, на рис. 3 во второй момент сразу три точки решили, что они верхние левые, и запустили огибающий сигнал (обозначается клеткой с одной диагональной линией). В какой-то момент каждый из этих сигналов может понять, что он не является настоящим огибающим сигналом, и тогда нужно запустить стирающий сигнал (изображается клеткой с двумя диагональными линиями), который сотрет ложный огибающий сигнал. Если скорость огибающего сигнала такая же, что и у стирающего сигнала, то стирающий сигнал не догонит огибающий и ложный огибающий сигнал дойдет до верней левой точки или до ложной верней левой точки, так и не поняв, что он ложный. Например, так должно было бы случиться с ложным огибающим сигналом, выпущенным из начала нижней запятой, если бы стирающий сигнал его не догнал. Музаффарова в [22] доказала, что если скорость огибающего сигнала будет в 2 раза меньше скорости стирающего сигнала, то предложенный алгоритм нахождения верхней левой ячейки построенной выпуклой оболочки будет работать корректно.

Так, на рис. 3 показано, как строится выпуклая оболочка, как возникают и движутся с половинной скоростью три ложных огибающих сигнала, как возникают стирающие сигналы и как они движутся с единичной скоростью по следу огибающего сигнала и стирают его. В момент 12 возникает огибающий сигнал, который не будет ложным и благополучно вернется в верхнюю левую точку, после чего по верхней стороне запустится влево сигнал назначения командира, который, добежав до первой бывшей черной клетки, назначит эту клетку командиром. Состояния клеточного автомата после 12 такта на рисунке не изображены.

Приведем еще один пример задачи. **Задача построения кратчайшего пути** для клеточных автоматов звучит следующим образом. В начальной конфигурации имеются только две ячейки в активном состоянии, которые назовем начальными точками. Кратчайший путь считается построенным, если с какого-то момента конфигурация становится стабильной и активные

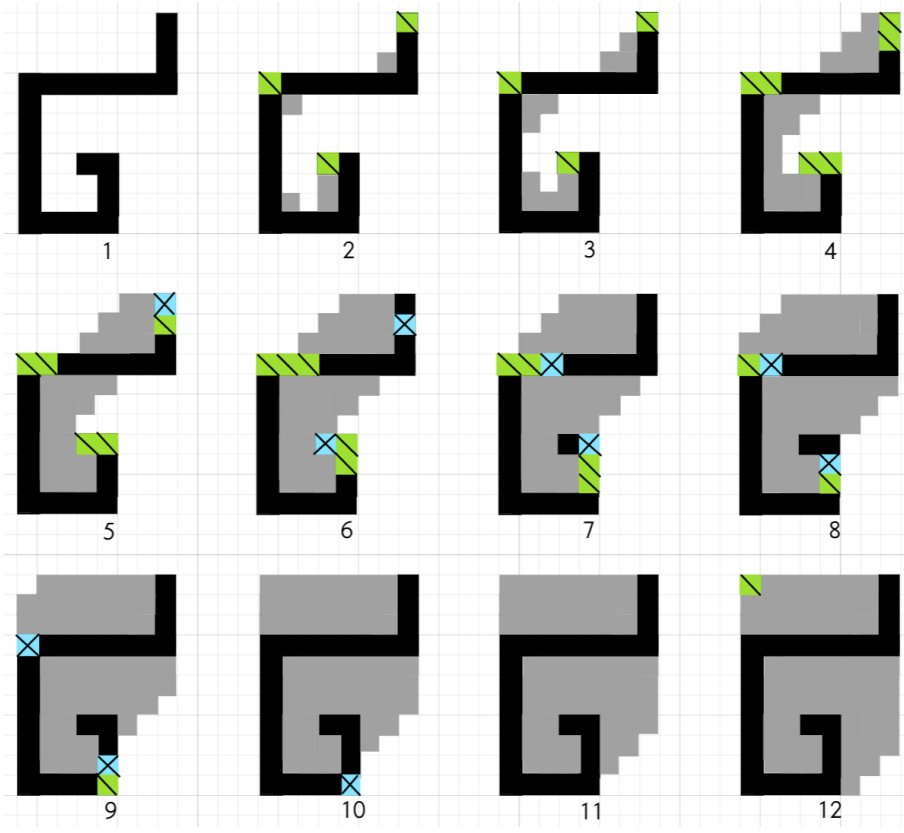

Рис. 3. Задача назначения командира

ячейки этой конфигурации образуют кратчайший в манхэттенской метрике путь между начальными точками.

В работе [29] приводится решение этой задачи клеточными автоматами, состоящее из трех этапов.

1. Распространение расширяющегося сигнала от одной из начальных точек. При расширении каждая ячейка запоминает, откуда к ней пришел сигнал. Это позволит в дальнейшем осуществить обратный ход.
2. Когда волна достигает второй начальной точки, осуществляется обратный ход, приводящий к первой точке, который и дает кратчайший путь.
3. Одновременно запускается волна очищения (приведения в состояние покоя) всех ячеек, кроме ячеек пути. Чтобы эта волна догнала расширяющуюся волну, расширяющаяся волна с первого этапа должна расширяться с половинной скоростью, а волна третьего этапа — с единичной скоростью.

В работе [29] утверждается, что предложенный авторами автомат имеет 14 состояний. В этой работе не оценивается время построения пути,

но несложно понять, что оно не меньше, чем $6n$, где n — расстояние по Манхэттену между начальными точками. Здесь $2n$ тактов необходимо первому этапу и $4n$ тактов — третьему. Можно несколько ускорить процесс, пустив расширяющуюся волну из обоих начальных точек, но понятно, что все равно время построения пути будет пропорционально расстоянию между начальными точками. Более того, таким способом можно решать задачу построения минимального стягивающего дерева для большего, чем два, количества точек.

Представляет интерес **одномерная задача поиска ближайшего соседа**. Постановка задачи: на прямой одна клетка окрашена в черный цвет и некоторое количество клеток окрашено в красный цвет; нужно найти ближайшую к черной клетку красного цвета и все клетки между черной и найденной красной клеткой окрасить в оранжевый цвет (провести дорожку до ближайшего соседа); все остальные клетки в финальной конфигурации должны быть окрашены в белый цвет (цвет состояния покоя). В общем случае клеточным автоматом эту задачу решить нельзя, поскольку в обе стороны надо будет послать стирающий сигнал, который сотрет все красные клетки, но этот сигнал будет уходить в бесконечность, поскольку мы не знаем, как далеко может находиться красная клетка.

Задачу можно решить в ослабленной постановке, если отказаться от требования, чтобы в финальной конфигурации не было других клеток, кроме черной, ближайшей красной и оранжевых клеток, соединяющих черную и ближайшую красную клетки. В ослабленной постановке задачу можно решить за время пропорциональное расстоянию от черной клетки до ближайшей красной. Идея решения состоит в том, чтобы послать из черной клетки в обе стороны расширяющийся сигнал. Когда расширяющийся сигнал достигает черной клетки, он отражается и идет обратно. С какой стороны отраженный сигнал быстрее вернется к черной клетке, с той стороны и находится ближайший сосед, к которому и проводится оранжевая дорожка. Можно наоборот посылать расширяющийся сигнал из всех красных клеток в обе стороны. С какой стороны к черной клетке придет расширяющийся сигнал, в ту сторону и строится оранжевая дорожка.

С помощью клеточных автоматов можно выполнять арифметические и векторные операции. Сформулируем **задачу сложения векторов на прямой**.

Пусть в пространстве $\mathbb{Z}^1$ задано начальное состояние I клеточного автомата, удовлетворяющее следующим условиям:

1. Любой ячейке присвоено одно из трех состояний $\{B, E, *\}$, где $*$ соответствует состоянию покоя, B интерпретируется как начало координат, а E — как концы векторов.
2. Есть лишь одна ячейка, которой присвоено состояние B, и не больше двух ячеек, которым присвоено состояние E.

Решением задачи сложения векторов на прямой, соответствующей начальному состоянию I, назовем состояние автомата, называемое финальным и удовлетворяющее следующим условиям:

1. Ячейке, которой в начальном состоянии было присвоено состояние B, присвоено состояние B_F.

2. Ячейке, которая находится от ячейки B на расстоянии равном сумме векторов, присвоено состояние E_F.
3. Остальным ячейкам присвоено состояние $*$.
4. Описанное выше финальное состояние в дальнейшем не изменяется.

Иными словами, в начальной конфигурации активными ячейками являются начало координат и концы двух векторов, а в конечной конфигурации остается начало координат и один конец вектора, равного сумме исходных векторов.

Сразу отметим, что если в начальном состоянии могут быть как одна, так и две ячейки в состоянии E, то начало координат (ячейка в состоянии B) никогда не сможет понять, имеются одна или две ячейки в состоянии E, также, как и ячейка E не сможет понять, есть ли еще ячейка в состоянии E, поскольку непонятно, сигнал от второй ячейки E еще не дошел или он никогда не дойдет. Поэтому в дальнейшем будем предполагать, что в начальном состоянии всегда есть две ячейки в состоянии E.

Приведем решение этой задачи, состоящее из трех этапов.

На первом этапе начало координат и концы векторов (ячейки в состояниях E) стараются понять взаимное расположение относительно друг друга. Для этого концы векторов посылают в обе стороны расширяющиеся сигналы со скоростью $1/2$. Расширяющиеся сигналы, достигнув начала координат, гаснут. Когда до начала координат доходят оба расширяющихся сигнала, начало координат понимает, с какой стороны от него находятся концы векторов.

Если расширяющиеся сигналы приходят к началу координат с разных сторон, то начало координат понимает, что концы векторов находятся по разные стороны от начала координат. Более того, начало координат понимает, с какой стороны вектор длиннее. Тогда начало координат посылает влево сигнал «ты единственная слева и длиннее», если левый вектор длиннее, и сигнал «ты единственная слева и короче», если левый вектор короче. Вправо начало координат посылает сигнал «ты единственная справа».

Левая ячейка E, получив сигнал «ты единственная слева и длиннее» или сигнал «ты единственная слева и короче», понимает, что она слева от начала координат и что вторая ячейка E находится справа от начала координат. Более того, она понимает, с какой стороны от начала координат будет результирующий вектор суммы. Сигнал «ты единственная слева и длиннее» или «ты единственная слева и короче», перевалив через конец вектора, превращается в стирающий сигнал, догоняет движущийся влево расширяющийся сигнал и гасит его. Аналогично правая ячейка E, получив сигнал «ты единственная справа», понимает, что она справа от начала координат и что вторая ячейка E находится слева от начала координат. Аналогично сигнал «ты единственная справа», перевалив через конец вектора, превращается в стирающий сигнал, догоняет движущийся вправо расширяющийся сигнал и гасит его. В результате концы векторов и начало координат понимают свое взаимное расположение и расширяющиеся сигналы через какое-то время исчезнут.

Предположим теперь, что оба расширяющихся сигнала пришли к началу координат справа (случай, когда оба слева, рассматривается аналогично). Тогда начало координат посылает вправо сигнал «ты справа». Когда

этот сигнал достигает первой ячейки в состоянии E, эта ячейка понимает, что она первая справа. Перевалив через эту ячейку, сигнал «ты справа» превращается в сигнал «ты вторая справа» и бежит дальше вправо. Когда он достигает второй ячейки в состоянии E, эта ячейка понимает, что она вторая справа. А сигнал «ты вторая справа», перевалив через эту ячейку, превращается в стирающий сигнал, который догоняет распространяющийся вправо сигнал и гасит его. Таким образом, и в этом случае концы векторов и начало координат понимают свое взаимное расположение.

Обозначим левый конец вектора через L, а правый — через R. На втором этапе мы найдем середину отрезка L, R, или, что то же самое, полусумму векторов L и R. Для этого запустим из ячейки L два сигнала: один со скоростью 1, а второй со скоростью $1/3$. Первый сигнал, достигнув ячейки R, отражается и возвращается назад. Точка встречи двух сигналов и будет серединой отрезка L, R. Обозначим эту ячейку через S. Поскольку ячейка L знала, с какой стороны от начала координат будет результирующий вектор, то можно считать, что и ячейка S знает, с какой стороны от начала координат она находится.

На третьем этапе мы будем удваивать вектор S. Для этого мы от ячейки S в сторону начала координат пошлем сигнал со скоростью 1, а в обратную сторону — сигнал со скоростью $1/3$. Первый сигнал, достигнув начала координат, отражается и возвращается назад. Точка встречи двух сигналов и будет результирующим вектором суммы.

Можно найти более быстрый алгоритм сложения векторов, но понятно, что в любом случае время вычисления будет не меньше, чем длина большего вектора.

Теперь рассмотрим **задачу однонаправленного движения точки на луче**, которая исследовалась в работах Е. Е. Титовой [26]. Эта задача состоит в следующем.

Множество ячеек представляет собой множество натуральных чисел, т. е. луч, направленный вправо. Каждая ячейка имеет двух соседей, одного слева и одного справа. Часть состояний ячеек называются метками и считаются черными, остальные состояния считаются белыми. Правильными считаются конфигурации, когда на луче ровно одна черная ячейка, называемая точкой. Ячейка, соответствующая числу 1 (самая левая ячейка), не имеет соседа слева, и переменную, соответствующую состоянию соседа слева этой ячейки, будем воспринимать как управляющий вход, на который можем подавать любые управляющие воздействия. Описанное множество ячеек с одним управляющим входом будем называть *экраном*.

С формальной точки зрения *экраном* называется клеточный автомат $S = (\mathbb{N}, E_n, V = \{-1, 1\}, \varphi, M)$, где n — число состояний ячейки клеточного автомата, а $\varphi : E_n^3 \to E_n$ — локальная функция переходов, M — множество меток, $M \subset E_n$, $0 \notin M$. Если ячейка находится в состоянии из M, то неформально считаем, что она окрашена в черный цвет, иначе она окрашена в белый цвет. Переменную x_{-1} локальной функции переходов ячейки, соответствующей числу 1 (*самой левой ячейки*), назовем *управляющим входом экрана* S.

Законом движения назовем бесконечную последовательность (сверхслово) из нулей и единиц. Если $F = f_1, f_2, f_3, \ldots$ — закон движения, то через $F(t)$ обозначим t-й элемент последовательности, т. е. $F(t) = f_t$.

Будем говорить, что *на экране S реализуется движение точки по закону F*, если выполняются следующие условия:

1) в некоторый момент времени в самой левой ячейке экрана появляется метка (до этого на экране нет меток), этот момент будем называть *моментом начала движения* или *началом движения*;
2) изменение позиции метки на экране в t-й момент от начала движения соответствует t-й букве в сверхслове F, а именно если $F(t) = 0$, то в $(t+1)$-й момент метка остается в той же ячейке, где была в текущий момент; если $F(t) = 1$, то в $(t+1)$-й момент метка сдвинется на одну ячейку вправо по сравнению со своим текущим положением;
3) в каждый момент времени после начала движения на экране есть ровно одна метка.

Экран S будем называть *универсальным для множества законов движения $\mathcal{F}$*, если для любого F из $\mathcal{F}$ существует такая управляющая последовательность, подаваемая на управляющий вход экрана, что на экране реализуется движение точки по закону F .

Через $\mathcal{F}^s$ обозначим множество таких законов движения F, в которых не встречается более чем s единиц подряд.

В работе [26] доказаны следующие теоремы

Теорема 1 (Е. Е. Титова [26]). *Для любого экрана S существует такой закон движения $F \in \{0,1\}^\infty$, что на экране S невозможно реализовать движение точки по закону F.*

Теорема 2 (Е. Е. Титова [26]). *Существует закон движения $F \in \{0,1\}^\infty$, движение по которому невозможно реализовать ни на каком экране S.*

Теорема 3 (Е. Е. Титова [26]). *Существует универсальный экран с $2s+2$ состояниями для множества законов движения $\mathcal{F}^s$.*

Вопрос описания множества всех реализуемых законов движения остается открытым, хотя в работе Г. В. Калачева и Е. Е. Титовой [13] сделаны существенные продвижения в этом направлении.

В работах Е. В. Кузнецовой [18, 19] рассматривается двунаправленное движение на луче. В этой задаче в законе движения могут встречаться символы -1, которые интерпретируются как движение назад. В работе [18] показано, что существует универсальный экран с 5 состояниями для законов движения со скоростью движения вперед не более $1/2$ и произвольной скоростью движения назад и что универсального автомата с меньшим числом состояний для этого класса законов движения не существует. В работе [19] исследуется вопрос, какие классы законов движения можно реализовать на экране с 4 состояниями. Но многие вопросы двунаправленного движения на луче остаются открытыми.

§ 3. Клеточные автоматы с локаторами

Из приведенных выше примеров мы видим, что одним из серьезных ограничений клеточных автоматов является ограниченность шаблона соседства, т. е. каждый автомат может видеть некоторое число своих соседей

и тем самым сигналы в клеточных автоматах распространяются относительно медленно. Это приводит у тому, что время решения задачи, как правило, пропорционально линейным размерам начальной конфигурации. С этим же связана сложность синхронизации работы отдельных элементарных автоматов, откуда проистекает сложность управления процессом решения. Поэтому хочется добавить клеточным автоматам возможность передавать некоторые сигналы всем элементарным автоматам одновременно, что позволит преодолеть свойство локальности, устранит проблему синхронизации и улучшит управляемость процессом решения задач.

Здесь можно вспомнить модель несжимаемой жидкости, в которой тоже сигналы моментально распространяются по всему объему. Похожая картина наблюдается в квантовой механике и в квантовых клеточных автоматах [27], когда изменение состояния одного автомата вызывает изменение состояния всех «запутанных» с ним автоматов. В работе [30] вводится понятие нелокальных клеточных автоматов. В этой работе нелокальность состоит в том, что для каждого элементарного автомата множество его соседей выбирается случайным образом и, таким образом, соседними могут оказаться далеко отстоящие друг от друга элементарные автоматы.

В обычной жизни человек, когда хочет передать информацию не только видимым соседям, может воспользоваться такими приемами, как подача световых сигналов с помощью сигнальных ракетниц. Еще более распространенным способом является использование радио- и телеэфира.

В данной работе тоже вводится понятие эфира. Считается, что каждый элементарный автомат может послать в эфир некоторый сигнал из конечного алфавита. Элементы алфавита образуют конечную аддитивную коммутативную полугруппу, а сам эфир представляет собой потенциально бесконечный сумматор сигналов элементарных автоматов, где в качестве суммы выступает определяющая операция данной полугруппы. На следующий такт каждый элементарный автомат получает из эфира суммарный сигнал и, учитывая его, изменяет свое состояние. В природе таким сумматором является эфир, который суммирует все радиосигналы естественным образом, и фактически каждый из приемников получает на вход один и тот же сигнал и уже потом выделяет из общего сигнала нужную ему составляющую.

На этом принципе можно реализовать новый тип интегральных схем, используя в качестве сумматора некоторую подложку, на которую все элементарные автоматы будут сбрасывать некоторые коммутирующие или экстренные сигналы.

Введение эфира и возможности посылать в эфир сигналы позволяет мгновенно передавать сигналы на любые расстояния и тем самым позволяет одному элементарному автомату управлять поведением сколь угодно далеко удаленного от него другого элементарного автомата. Рассматриваются клеточные автоматы с локаторами, которые могут получать сигналы из эфира с определенных направлений. Иными словами, у каждого элементарного автомата имеется несколько локаторов, направленных в разные стороны, и он может с помощью этих локаторов получать сигналы с этих направлений.

Далее мы введем формальную модель клеточных автоматов с локаторами, приведем решение описанных выше задач и покажем, насколько они проще решаются с помощью клеточных автоматов с локаторами. Кроме

того, опишем как с помощью клеточных автоматов с локаторами можно реализовать базы данных типа «ключ—значение», тогда как для клеточных автоматов адекватного решения этой задачи не существует.

§ 4. Понятие клеточного автомата с локаторами

В работе Э. Э. Гасанова [6] введено понятие клеточного автомата с локаторами. В работе Г. В. Калачева [11] были выявлены некоторые неточности приведенного в [6] определения. Наиболее точное определение клеточного автомата с локаторами приводится в работе Д. Э. Ибрагимовой [10], и мы воспользуемся этим определением с минимальными изменениями.

Под *телесным углом* в $\mathbb{R}^k$ будем понимать часть пространства $\mathbb{R}^k$, которая является объединением всех лучей, выходящих из данной точки (*вершины угла*) и пересекающих некоторую гиперповерхность в $\mathbb{R}^k$. По определению будем считать, что вершина телесного угла не входит в телесный угол.

Рациональным телесным углом будем называть телесный угол, границы которого являются частями гиперплоскостей, задаваемых линейными уравнениями с целыми коэффициентами.

В частности, в данной работе мы в основном будем рассматривать два вырожденных случая: *полный телесный угол*, совпадающий с $\mathbb{R}^k$ без вершины угла, который будем обозначать через Ω, и телесные углы, равные одному лучу, такие телесные углы будем обозначать через векторы, являющиеся направляющими лучей.

Клеточным автоматом с локаторами называется восьмерка $\sigma = (\mathbb{Z}^k, Q, V, G, +, L, \varphi, \psi)$, где $\mathbb{Z}^k$ — множество k-мерных векторов с целыми координатами, Q — некоторое конечное множество, называемое *множеством состояний*; в множестве Q выделено одно состояние q_0, называемое *состоянием покоя*; $V = (\alpha_1, \ldots, \alpha_{h-1})$ — упорядоченный набор попарно различных векторов из $\mathbb{Z}^k$; G — коммутативная полугруппа с нейтральным элементом e; $+$ — коммутативная полугрупповая операция, заданная на G; $L = (\nu_1, \ldots, \nu_m)$ — упорядоченный набор попарно различных рациональных телесных углов в $\mathbb{R}^k$ с вершиной в начале координат; φ — функция, зависящая от переменных $x_0, x_1, \ldots, x_{h-1}, z_1, \ldots, z_m$; $\varphi : Q^h \times G^m \to Q$, $\varphi(\boldsymbol{q_o}, \boldsymbol{e}) = q_0$; $\boldsymbol{q_o} = (q_0, \ldots, q_0) \in Q^h$, $\boldsymbol{e} = (e, \ldots, e) \in G^m$; ψ — функция, зависящая от переменных $x_0, x_1, \ldots, x_{h-1}, z_1, \ldots, z_m$; $\psi : Q^h \times G^m \to G$; $\psi(\boldsymbol{q_o}, \boldsymbol{e}) = e$. Элементы множества $\mathbb{Z}^k$ называются *ячейками* клеточного автомата σ; элементы множества Q называются *состояниями ячейки* клеточного автомата σ; набор V называется *шаблоном соседства* клеточного автомата σ; элементы множества G называются *сигналами вещания*; набор L называется *шаблоном локаторов* клеточного автомата σ; функция φ называется *локальной функцией переходов* автомата σ; функция ψ называется *функцией вещания* автомата σ; переменные $x_0, x_1, \ldots, x_{h-1}$ принимают значения из Q, переменные $z_1, \ldots, z_m$ принимают значения из G. Состояние q_0 интерпретируется как *состояние покоя*, а условие $\varphi(\boldsymbol{q_o}, \boldsymbol{e}) = q_0$ — как *условие сохранения состояния покоя*. Ячейки, находящиеся в состоянии отличном от q_0, будем называть *активными*. Условие $\psi(\boldsymbol{q_o}, \boldsymbol{e}) = e$ означает, что ячейка в состоянии покоя, не имеющая активных соседей и не получающая сигналов из эфира,

посылает в эфир нейтральный элемент, что можно интерпретировать как то, что она не посылает сигналы в эфир.

Здесь нам нужно было вводить упорядочение шаблона соседства V и шаблона локаторов L для того, чтобы установить взаимно-однозначное соответствие между векторами из V и телесными углами из L и переменными локальной функции переходов φ и функции вещания ψ соответственно $x_0, x_1, \ldots, x_{h-1}$ и $z_1, \ldots, z_m$. Это соответствие можно сделать более явным, если индексировать переменные функций φ и ψ самими векторами и телесными углами, т. е. считать, что локальная функция переходов φ и функции вещания ψ зависят от переменных $x_{\mathbf{o}}, x_{\alpha_1}, \ldots, x_{\alpha_{h-1}}, z_{\nu_1}, \ldots, z_{\nu_m}$, здесь индекс первой переменной есть нулевой вектор $\mathbf{o} = (0, \ldots, 0) \in \mathbb{Z}^k$. Если договориться так индексировать переменные локальной функции переходов и функции вещания, то их можно записывать в любом порядке и тогда можно воспринимать шаблон соседства и шаблон локаторов как просто множества, а не упорядоченный набор. В дальнейшем мы будем индексировать переменные локальной функции переходов и функции вещания векторами из шаблона соседства и телесными углами из шаблона локаторов.

При этом мы часто будем опускать в индексах внешние круглые скобки у векторов. Например, если $k = 2$, $n = 2$, $q = 2$ и $V = \{(-1, 0), (1, 0)\}$, $L = \{\Omega, (0, 1)\}$, то пример локальной функции переходов может выглядеть так: $\varphi = x_{-1,0} \& z_\Omega \vee x_{1,0} \& z_{0,1}$.

Если $\alpha \in \mathbb{Z}^k$ и ν — телесный угол с вершиной в начале координат, то через $\nu(\alpha)$ обозначим телесный угол, полученный параллельным переносом телесного угла ν на вектор α, т. е. вершиной телесного угла $\nu(\alpha)$ является точка α.

Если $\alpha \in \mathbb{Z}^k$ — ячейка клеточного автомата σ, то множество $V(\alpha) = \{\alpha, \alpha+\alpha_1, \ldots, \alpha+\alpha_{h-1}\}$ называется *окрестностью ячейки* α, а множество $L(\alpha) = \{\nu_1(\alpha), \ldots, \nu_m(\alpha_m)\}$ называется *локаторами ячейки* α.

Состоянием клеточного автомата с локаторами σ назовем пару (g, f), где g — произвольная функция, определенная на множестве Z^k, принимающая значения из G, называемая *состоянием эфира*, f — произвольная функция, определенная на множестве Z^k, принимающая значения из Q и называемая *распределением состояний клеточного автомата с локаторами* σ. Такую пару функций можно интерпретировать как некую мозаику, получающуюся в k-мерном пространстве приписыванием каждой точке с целочисленными координатами некоторого сигнала из G и некоторого состояния из Q. Множество всевозможных состояний клеточного автомата с локаторами обозначим Σ.

Если $\alpha \in \mathbb{Z}^k$, (g, f) — состояние клеточного автомата с локаторами σ, то значение $g(\alpha)$ назовем *сигналом ячейки* α, определяемым состоянием (g, f), а значение $f(\alpha)$ — *состоянием ячейки* α, определяемым состоянием (g, f).

Для каждого $i \in \{1, \ldots, m\}$

$$s_i(\alpha) = \sum_{\beta \in \nu_i(\alpha) \cap \mathbb{Z}^k} g(\beta) \tag{2}$$

назовем *значением локатора* ν_i, определяемым состоянием (g, f). Здесь суммирование сигналов осуществляется с помощью определяющей операции $+$ полугруппы G. Отметим, что в формулах (2) используются формально

бесконечные суммы, и, чтобы они были определены, мы либо будем считать, что только конечное число слагаемых в суммах отлично от нейтрального элемента, либо предположим, что полугруппа $(G, +)$ является идемпотентным моноидом, т. е. для любого $h \in G$ выполнено $h+h=h$.

На множестве Σ определим *глобальную функцию переходов* Φ_σ клеточного автомата с локаторами σ, полагая $\Phi_\sigma(g, f) = (g', f')$, где $(g, f), (g', f') \in \Sigma$ и для любой ячейки $\alpha \in \mathbb{Z}^k$ выполняются тождества

$$f'(\alpha) = \varphi(f(\alpha), f(\alpha+\alpha_1), \ldots, f(\alpha+\alpha_{h-1}), s_1(\alpha), \ldots, s_m(\alpha)), \tag{3}$$

$$g'(\alpha) = \psi(f(\alpha), f(\alpha+\alpha_1), \ldots, f(\alpha+\alpha_{h-1}), s_1(\alpha), \ldots, s_m(\alpha)). \tag{4}$$

Содержательная интерпретация отображения Φ_σ такова, что сигнал каждой ячейки и состояние каждой ячейки «после перехода» определяется по состоянию упорядоченной окрестности ячейки и по значениям локаторов «до перехода» с помощью законов φ и ψ одинаково для всех ячеек.

Поведениями клеточного автомата с локаторами σ назовем такие последовательности $(g_0, f_0), (g_1, f_1), (g_2, f_2), \ldots$ его состояний, для которых выполняется $(g_{i+1}, f_{i+1}) = \Phi_\sigma(g_i, f_i)$ для всех $i = 0, 1, 2, \ldots$, причем (g_i, f_i) называется *состоянием клеточного автомата с локаторами* σ *в момент* i, а (g_0, f_0) называется *начальным состоянием клеточного автомата с локаторами* σ.

Состояние клеточного автомата, у которого лишь конечное число ячеек находится в отличном от состояния покоя g_0 и сигналы лишь конечного числа ячеек не равны нейтральному элементу e, назовем *конфигурацией*. Множество конфигураций будем обозначать через Σ'.

Вернемся к рассмотренным ранее задачам для клеточных автоматов и покажем, что эти задачи клеточными автоматами с локаторами решаются существенно проще.

§ 5. Задача синхронизации стрелков

Клеточный автомат с локаторами σ_0, который решает задачу синхронизации стрелков, имеет следующий вид: $\sigma_0 = (\mathbb{Z}^1, E_3, V = \{-1, 1\}, E_2, \vee, L = \{\Omega\}, \varphi, \psi)$, где $\vee$ — дизъюнкция, взятая в качестве определяющей операции на полугруппе сигналов $E_2 = \{0, 1\}$, шаблон локаторов состоит из одного полного телесного угла Ω, функция вещания ψ принимает значение 1 только для самого левого солдата в начальном состоянии, т. е. $\psi(x_0, x_{-1}, x_1, z_\Omega) = x_0 \bar{x}_{-1} \bar{z}_\Omega$, локальная функция переходов принимает значение 2, только если ячейка находится в состоянии 1 и сигнал эфира равен 1, и не меняет состояния во всех остальных случаях, т. е. $\varphi(x_0, x_{-1}, x_1, z_\Omega) = \max(2 \cdot ((x_0 = 1) \& (z_\Omega = 1) \vee (x_0 = 2)), 1 \cdot (x_0 = 1))$.

Тем самым задачу синхронизации стрелков можно решить за 2 такта с помощью трех состояний и двух сигналов вещания. И фактически это решение полностью соответствует решению, принятому среди военных: командир (самый левый солдат) командует «огонь» и вся команда стреляет.

§ 6. Однонаправленное движение точки на луче

Приведем клеточный автомат с локаторами, который решает задачу однонаправленного движения точки на луче.

Рассмотрим следующий клеточный автомат с локаторами $\sigma_1 = (\mathbb{N}, E_2, V = \{-1, 1\}, E_2, \vee, L = \{\Omega\}, \varphi, \psi, M = \{1\})$, где $\vee$ — дизъюнкция, взятая в качестве определяющей операции на полугруппе сигналов $E_2 = \{0, 1\}$, шаблон локаторов состоит из одного полного телесного угла Ω, множество меток состоит из одного символа 1, функция вещания ψ тождественно нулевая, локальная функция переходов принимает значение 1 только в двух случаях: если ячейка находится в состоянии 1 и сигнал эфира равен 0 или если ячейка слева находится в состоянии 1 и сигнал эфира равен 1, т. е. $\varphi(x_0, x_{-1}, x_1, z_\Omega) = x_0 \& \bar{z}_\Omega \vee x_{-1} \& z_\Omega$.

Будем считать, что переменная x_{-1} локальной функции переходов самой левой ячейки является управляющим входом. Кроме того, будем считать, что в качестве управляющих воздействий можно посылать в эфир сигналы из E_2.

Легко видеть, что, чтобы начать движение, надо на управляющий вход подать 1, а также послать 1 в эфир. В результате на экране в самой левой ячейке появится метка. Далее, чтобы реализовать закон движения F, надо в момент t посылать в эфир значение $F(t)$.

Тем самым с помощью клеточных автоматов с локаторами можно реализовать любой закон движения, причем число состояний этого автомата равно 2 и мощность алфавита вещания равна 2.

Фактически с помощью сигналов в эфир мы даем команды точке, двигаться ей или стоять.

Понятно, что аналогичным образом решается и задача двунаправленного движения на луче.

§ 7. Построение кратчайшего пути

Рассмотрим решение задачи построения кратчайшего пути клеточными автоматами с локаторами.

Рассмотрим следующий клеточный автомат с локаторами: $\sigma_2 = (\mathbb{Z}^2, E_2, V = \{(-1, 0), (0, 1), (1, 0), (0, -1)\}, E_2, \vee, L = \{(-1, 0), (0, 1), (1, 0), (0, -1)\}, \varphi, \psi)$, где $\vee$ — дизъюнкция, взятая в качестве определяющей операции на полугруппе сигналов $E_2 = \{0, 1\}$, шаблон соседства «крест», шаблон локаторов состоит из четырех лучей, направленных влево, вверх, вправо и вниз, функция вещания ψ принимает значение 1, если ячейка в состоянии 1 и имеет место один из четырех случаев: если все ее соседи в состоянии 0; у нее нет соседа сверху в состоянии 1 и верхний локатор получает сигнал 1; у нее нет соседа слева в состоянии 1 и левый локатор получает сигнал 1; у нее нет соседа справа в состоянии 1 и правый локатор получает сигнал 1; т. е.

$$\psi(x_0, x_{-1,0}, x_{0,1}, x_{1,0}, x_{0,-1}, z_{-1,0}, z_{0,1}, z_{1,0}, z_{0,-1}) = \\ = x_0(\bar{x}_{-1,0}\bar{x}_{0,1}\bar{x}_{1,0}\bar{x}_{0,-1} \vee \bar{x}_{0,1}z_{0,1} \vee \bar{x}_{-1,0}z_{-1,0} \vee \bar{x}_{1,0}z_{1,0});$$

локальная функция переходов принимает значение 1, если ячейка была в состоянии 1 или если одновременно пришли сигналы из эфира на одну из четырех пар локаторов: верхний и правый, верхний и левый, верхний и нижний, левый и правый, т. е.

$$\varphi(x_0, x_{-1,0}, x_{0,1}, x_{1,0}, x_{0,-1}, z_{-1,0}, z_{0,1}, z_{1,0}, z_{0,-1}) = \\ = x_0 \vee z_{0,1}z_{1,0} \vee z_{0,1}z_{-1,0} \vee z_{0,1}z_{0,-1} \vee z_{-1,0}z_{1,0}.$$

Покажем, что приведенный выше клеточный автомат с локаторами решает задачу построения кратчайшего пути.

В начальный (нулевой) такт на плоскости только две активные ячейки, которые называем начальными.

Рассмотрим различные случаи расположения начальных ячеек.

Случай 1. Начальные ячейки расположены на одной горизонтали. Левую начальную ячейку обозначим через A, а правую — через B.

Случай 1.1. Если начальные ячейки соседние, то эта пара ячеек и составляет кратчайший путь. Осталось заметить, что сигналы в эфир не появятся, следовательно, не появятся новые активные ячейки и, значит, конфигурация останется стабильной.

Случай 1.2. Если начальные ячейки не соседние, то функция вещания каждой из начальных ячеек станет равной 1, поскольку у этих ячеек нет соседей. Следовательно, на такте 1 в эфир от ячеек A и B пойдет сигнал и для всех ячеек между ячейками A и B левые и правые локаторы получат сигналы. Значит, локальные функции переходов этих ячеек примут значение 1. Следовательно, на такте 2 все ячейки между A и B перейдут в состояние 1. Кратчайший путь построен. Поскольку у всех ячеек есть соседи, функция вещания всех ячеек примет значение 0 и возникшая конфигурация останется стабильной.

Случай 2. Начальные ячейки расположены на одной вертикали. Этот случай доказывается аналогично случаю 1.

Случай 3. Начальные ячейки находятся в общем положении. Мысленно проведем через начальные ячейки вертикальные и горизонтальные линии. Получаем воображаемый прямоугольник со сторонами, параллельными осям координат, в двух диагональных вершинах которого расположены начальные ячейки.

Случай 3.1. Одна начальная ячейка расположена в левом верхнем углу прямоугольника (обозначим ее через A), а вторая — в правом нижнем (обозначим ее через B).

Поскольку у начальных ячеек нет соседей, то функция вещания этих ячеек примет значение 1. Следовательно, на такте 1 в эфир от ячеек A и B пойдет сигнал. Ячейка, расположенная в левом нижнем углу прямоугольника (обозначим ее через C), получит сигналы от верхнего и правого локаторов. Поэтому ее локальная функция переходов примет значение 1. Значит, на такте 2 ячейка C станет активной.

Случай 3.1.1. Ячейка C является соседней и для A, и для B. Следовательно, кратчайший путь построен. У всех ячеек есть соседи, следовательно, сигналы в эфир больше подаваться не будут и конфигурация останется стабильной.

Случай 3.1.2. Ячейка C не является соседней для A и является соседней для B. Тогда функция вещания ячейки C примет значение 1. Следовательно, на такте 3 в эфир пойдут сигналы от двух ячеек: A и C. Значит, все

ячейки между A и C получат сигналы на свои верхние и нижние локаторы. Следовательно, их локальная функция переходов примет значение 1. Следовательно, на такте 4 все ячейки между A и C станут активными. Теперь у всех ячеек есть соседи, следовательно, сигналы в эфир больше подаваться не будут и конфигурация останется стабильной.

Случай 3.1.3. Ячейка C является соседней для A и не является соседней для B. Этот случай доказывается аналогично случаю 3.1.2.

Случай 3.1.4. Ячейка C не является соседней ни для A, ни для B. Тогда функция вещания ячейки C примет значение 1. Следовательно, на такте 3 в эфир пойдут сигналы от трех ячеек: A, B и C. Значит, все ячейки между A и C получат сигналы на свои верхние и нижние локаторы, а все ячейки между B и C получат сигналы на свои левые и правые локаторы. Следовательно, локальная функция переходов всех этих ячеек примет значение 1. Следовательно, на такте 4 все ячейки между A и C и между B и C станут активными. Кратчайший путь построен. У всех ячеек есть соседи, следовательно, сигналы в эфир больше подаваться не будут и конфигурация останется стабильной.

Случай 3.2. Одна начальная ячейка расположена в левом нижнем углу воображаемого прямоугольника, а вторая — в правом верхнем. Этот случай доказывается аналогично случаю 3.1.

Таким образом, мы показали, что предлагаемый клеточный автомат с локаторами позволяет построить кратчайший путь не более чем за 4 такта. При этом у него только 2 состояния и 2 сигнала вещания.

§ 8. Нахождение выпуклой оболочки

Задачу нахождения выпуклой оболочки можно решить клеточными автоматами с локаторами за 2 такта. Далее мы уже не будем формально описывать автоматы, решающие интересующие нас задачи, а будем давать неформальные описания алгоритмов на идейном уровне.

Клеточный автомат, решающий задачу нахождения выпуклой оболочки, будет иметь 4 локатора, направленных вверх, вниз, вправо и влево. Алфавит сигналов вещания есть полугруппа $(\{0,1\},\vee)$. В начальный момент все черные клетки посылают в эфир сигнал 1. На следующий такт все черные клетки окрашиваются в синий цвет (чтобы больше не посылать сигналы в эфир), а белые клетки, которые получили сигналы 1 хотя бы с двух локаторов, окрашиваются в серый цвет. Белые, серые и синие ячейки сигналы в эфир не посылают (посылают сигнал 0).

Выпуклая оболочка построена за 2 такта и состоит из серых и синих клеток.

§ 9. Назначение командира

Задача назначения командира решается клеточными автоматами с локаторами за 3 такта.

У клеточного автомата, решающего задачу назначения командира, будет 4 локатора, направленных вверх, вниз, вправо и влево. За 2 такта мы построим выпуклую оболочку, как описано выше.

Во второй такт все верхние синие ячейки (те, у которых сверху белый сосед) посылают в эфир сигнал «я сверху». Та верхняя синяя ячейка, которая услышала сигнал «я сверху» только с правого локатора, понимает, что она самая левая, и на третий такт окрашивается в красный цвет (становится командиром). Остальные синие клетки на третий такт становятся коричневыми (чтобы больше не посылать сигналов в эфир), а все серые клетки становятся белыми.

Здесь мы использовали прием, который позволяет активным клеткам, находящимся на одной горизонтальной или вертикальной линии, понимать свое положение относительно других активных клеток. Для этого все активные клетки посылаю в эфир некий сигнал вещания. Те клетки, которые получают этот сигнал только с правого локатора или не получают ни с правого, ни с левого, понимают, что они самые левые в горизонтальной линии. Те клетки, которые получают этот сигнал только с левого локатора или не получают вовсе, понимают, что они самые правые в горизонтальной линии. Аналогично клетка может понять, что она самая верхняя или самая нижняя в вертикальной линии.

Далее мы будем использовать этот прием, не вдаваясь в детали.

§ 10. Поиск ближайшего соседа

Решение этой задачи на прямой клеточными автоматами получено Д. И. Васильевым. В работе [2] он построил автомат, который решает задачу за логарифмическое от расстояния до ближайшего соседа время.

Заметим, что для всех предыдущих задач время решения не зависело от размеров начальной конфигурации, и это первая задача, где зависимость от размера есть.

Опишем алгоритм решения задачи поиска ближайшего соседа на прямой, предложенный в работе [2].

В предлагаемом алгоритме сигнал вещания трехкомпонентный: первая компонента принимает значения 0 или 1 и полугрупповая операция по этой компоненте — это сложение по модулю 2. Вторые две компоненты вспомогательные и полугрупповая операция по этим компонентам — это максимум.

На первом этапе, который называется этапом ориентации, все клетки по первой компоненте посылают сигнал 0 и игра идет только на вспомогательных компонентах. На первом такте черная клетка посылает в эфир сигнал «я центральная» по второй компоненте. Те красные клетки, которые слышат сигнал «я центральная» с правого локатора, понимают, что они слева от центральной, а те, которые слышат этот сигнал с левого локатора, понимают, что они справа от центральной. На следующий такт левые клетки определяют, какая клетка из них самая правая, а правые клетки определяют, какая клетка из них самая левая. Чтобы не путаться, левые клетки посылают сигнал в эфир по второй компоненте, а правые — по третьей. На третий такт все клетки, кроме центральной, самой левой и самой правой становятся белыми (переходят в состояние покоя), т. е. на третьем такте остаются активными центральная клетка и одна или две красные клетки. Оставшиеся красные клетки посылают в эфир сигнал «я осталась». И если каждая из них услышит этот сигнал, то они понимают, что их две.

Если осталась одна красная клетка, то она не услышит этот сигнал. Если центральная клетка слышит этот сигнал с двух сторон, то она понимает, что остались две красные клетки с разных сторон от центральной. Если сигнал «я осталась» приходит к центральной клетке только с одной стороны, то центральная клетка понимает, что осталась одна красная клетка.

Если осталась одна красная клетка, то центральная и красная клетки посылают в эфир сигнал «строим оранжевую дорожку». Все клетки, которые слышат этот сигнал с двух сторон, понимают, что они находятся между центральной и красной клеткой, и перекрашиваются в оранжевый цвет. Задача решена.

Если остались две красные клетки, то мы переходим ко второму этапу — этапу сравнения длин отрезков от центральной клетки к каждой из красных клеток.

До сих пор сигналы посылались только по вспомогательным компонентам, а по первой компоненте всегда посылался сигнал 0. Теперь первая компонента будет активно использоваться.

На первом такте второго этапа (т. е. на четвертом такте) центральная клетка посылает в эфир сигнал «я центральная» по второй компоненте, а красные клетки посылают в эфир сигнал «я красная» по третьей компоненте. Те клетки, которые в правый локатор слышат сигнал «я центральная», а в левый локатор — «я красная», понимают, что находятся между левой красной и центральной клеткой и становятся «левыми» (переходят в состояние q_L), а клетки, которые в левый локатор слышат сигнал «я центральная», а в правый локатор — «я красная», понимают, что находятся между центральной клеткой и правой красной и становятся «правыми» (переходят в состояние q_R). Центральная клетка, которая и будет формировать результат сравнения, переходит в состояние $q_C^=$, которое означает, что на данный момент мы думаем, что длины равные. И с этого момента начинается игра по сравнению количества «левых» и «правых» клеток, и эти количества будем называть длиной левого и правого отрезков соответственно.

В процессе сравнения длин все клетки в состоянии q_L и q_R посылают в эфир 1 по первой и второй компоненте. Остальные клетки посылают в эфир нейтральный элемент, т. е. нули по всем компонентам. Третья компонента эфира не существенна в процессе сравнения длин, и по ней всегда посылается нейтральный сигнал 0.

Идея алгоритма состоит в том, что на левом локаторе центральной ячейки формируется двоичное представление длины левого отрезка, начиная с младших бит, а правом локаторе центральной ячейки формируется двоичное представление длины правого отрезка, начиная с младших бит. Центральная ячейка сравнивает получающиеся числа следующим образом: если биты с левого и правого локаторов разные, то результат сравнения равен результату сравнения бит, если — одинаковые, то результат сравнения равен результату сравнения на предыдущем шаге. В начальный момент мы предположили, что длины равны.

Обозначим состояние покоя через $*$, состояние левой красной клетки через q_L^R, состояние правой красной клетки через q_R^R.

«Левые» клетки следят за сигналом эфира, поступающим только с левого локатора, и игнорируют правый локатор. «Правые» клетки, наоборот, игнорируют значения левого локатора и следят только за сигналами правого

локатора. Состояния в процессе сравнения длин изменяются по следующим правилам:

— клетка в состоянии q_L, которая с левого локатора получает сигнал 1 по первой компоненте, остается в состоянии q_L;

— клетка в состоянии q_R, которая с правого локатора получает сигнал 1 по первой компоненте, остается в состоянии q_R;

— центральная клетка, получающая с левого и правого локаторов одинаковые сигналы по первой и второй компоненте, остается в прежнем состоянии;

— центральная клетка, получающая на левый локатор 0 хотя бы по одной компоненте и получающая на правый локатор 1 по обоим компонентам, переходит в состояние $q_C^<$;

— центральная клетка, получающая на правый локатор 0 хотя бы по одной компоненте и получающая на левый локатор 1 по обоим компонентам, переходит в состояние $q_C^>$;

— клетки в состояниях q_L^R и q_R^R сохраняют свое состояние;

— во всех остальных случаях клетка переходит в нейтральное состояние $*$.

Через L_1, L_2 обозначим значения левого локатора по первой и второй компонентам, через R_1, R_2 — значения правого локатора по первой и второй компонентам и через Q — состояние ячейки в текущий момент.

Продемонстрируем алгоритм на примере, в котором будет 6 «левых» клеток и 5 «правых» клеток.

Таблица 1

$t = 5$

Q	$*$	q_L^R	q_L	q_L	q_L	q_L	q_L	q_L	$q_C^=$	q_R	q_R	q_R	q_R	q_R	q_R^R	$*$
L^1	0	0	0	1	0	1	0	1	0	0	1	0	1	0	1	1
L^2	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1
R^1	1	1	0	1	0	1	0	1	1	0	1	0	1	0	0	0
R^2	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0

Заметим, что в первый момент сравнения длин на левый и правый локаторы центральной клетки поступает младший бит двоичного представления длин левого и правого отрезков (т. е. четна или нечетна левая и правая длина). Далее отрезки вдвое прореживаются, т. е. фактически длина делится на два. После чего во второй момент мы получаем следующий по старшинству бит длин левого и правого отрезков и т. д. Процесс сравнения прекращается, когда одна или обе длины становятся равны 0 (чему соответствуют нули по второй компоненте на левом или правом локаторах центральной ячейки).

Таблица 2

$t = 6$

Q	$*$	q_L^R	$*$	q_L	$*$	q_L	$*$	q_L	$q_C^<$	$*$	q_R	$*$	q_R	$*$	q_R^R	$*$
L^1	0	0	0	0	1	1	0	0	1	1	1	0	0	1	1	1
L^2	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1
R^1	1	1	1	0	0	1	1	0	0	0	1	1	0	0	0	0
R^2	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0

Таблица 3

$t = 7$

Q	$*$	q_L^R	$*$	$*$	$*$	q_L	$*$	$*$	$q_C^>$	$*$	q_R	$*$	$*$	$*$	q_R^R	$*$
L^1	0	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0
L^2	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
R^1	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0	0
R^2	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0

Таблица 4

$t = 8$

Q	$*$	q_L^R	$*$	$*$	$*$	$*$	$*$	$*$	$q_C^>$	$*$	$*$	$*$	$*$	$*$	q_R^R	$*$
L^1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
L^2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R^1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R^2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Таким образом, когда на левом или правом локаторах центральной ячейки по второй компоненте появляется сигнал 0, то центральная ячейка понимает, что сравнение выполнено и результат сравнения отражен в ее состоянии. После этого остается только прорисовать оранжевую дорожку к более близкой красной клетке.

Отсюда мы получаем, что время решения задачи поиска ближайшего соседа не больше, чем логарифм от длины меньшего отрезка плюс небольшая константа (не большая чем 7).

В работе [4] Васильев и Гасанов показали, что решить эту задачу быстрее, чем за логарифмическое время, нельзя.

Приведем здесь основные идеи доказательства этой нижней оценки, но сначала более строго сформулируем задачу поиска ближайшего соседа на прямой. Пусть на $\mathbb{Z}$ задано начальное состояние $I = (g_0, f_0)$ клеточного автомата, удовлетворяющее следующим условиям:

1. Любой ячейке присвоено одно из трех состояний $\{q_S; q_C, *\}$, где $*$ — это состояние покоя, т. е. для любой ячейки $a \in \mathbb{Z}$ выполнено $f_0(a) \in \{q_S; q_C, *\}$.

2. Есть лишь одна ячейка, которой присвоено состояние q_C, и эта ячейка соответствует «центральной» точке.
3. Есть лишь конечное и непустое множество ячеек, которым присвоено состояние q_S и которые соответствуют красным точкам.
4. Все ячейки не посылают в эфир сигналы, т. е. для любой ячейки $a \in \mathbb{Z}$ выполнено $g_0(a) = e$.

Решением задачи поиска ближайшего соседа, соответствующей начальному состоянию I, назовем состояние $I_F = (g_F, f_F)$ автомата, удовлетворяющее следующим условиям:

1. Ячейке, которой в I было присвоено состояние q_C, присвоено состояние q_{CF}.
2. Ближайшей к ячейке в состоянии q_{CF} из тех, которым в I было присвоено состояние q_S, присвоено состояние q_{LE}, если она находится слева от ячейки в состоянии q_{CF}, и q_{RE} — если справа. Если таких ячеек две, то правой присваивается состояние $*$, а левой — q_{LE}.
3. Ячейкам, которые находятся между ячейками в состояниях q_{CF} и q_{LE}, присваивается состояние q_{LF}. Ячейкам, которые находятся между ячейками в состояниях q_{CF} и q_{RE}, присваивается состояние q_{RF}.
4. Остальные ячейки находятся в состоянии $*$.
5. Все ячейки не посылают в эфир сигналы, т. е. для любой ячейки $a \in \mathbb{Z}$ выполнено $g_F(a) = e$.

Скажем, что клеточный автомат с локаторами σ *решает задачу поиска ближайшего соседа*, если из любого начального состояния I, удовлетворяющего описанным выше условиям, он переходит в соответствующее I описанное выше финальное состояние I_F, и ни одно промежуточное состояние не содержит ячейки в состояниях q_{CF}, q_{LE}, q_{RE}, q_{LF} и q_{RF}.

Назовем *общим положением задачи поиска ближайшего соседа* задачу, в которой с обеих сторон от ячейки в состоянии q_C есть хотя бы по одной ячейке в состоянии q_S. Пусть T_I^σ — время, за которое произвольный автомат σ решает задачу поиска ближайшего соседа I (положим $T_I^\sigma = \infty$, если автомат σ не решает задачу I).

Теорема 4 (Д. И. Васильев, Э. Э. Гасанов [4]). *Для любого одномерного клеточного автомата с локаторами σ с мощностью алфавита вещания, равной M, и любого общего положения задачи поиска ближайшего соседа I выполнено $T_I^\sigma > \log_M(\frac{s-1}{6})$, где s — расстояние от ячейки в состоянии q_C до ближайшей ячейки в состоянии q_S в задаче I.*

Доказательство теоремы основывается на 4 леммах.

Скажем, что элемент h конечной полугруппы G имеет *предпериод* s ($s \in \mathbb{Z}$, $0 \leqslant s < |G|$) и *период* p ($p \in \mathbb{N}$, $1 \leqslant p \leqslant |G|$), если для любого $k \in \mathbb{N}$ выполнено $sh + kh = sh + (k+p)h$.

Лемма 1. *Для любого элемента h конечной произвольной полугруппы G существует предпериод s, $0 \leqslant s < |G|$, и период p, $1 \leqslant p \leqslant |G|$.*

Это достаточно очевидный факт. В самом деле, поскольку полугруппа G конечная, то среди элементов $h, 2h, 3h, \ldots, (|G|+1)h$ найдется два равных. Пусть это элементы k_1h и k_2h, $k_1<k_2$, $1\leqslant k_1\leqslant|G|$, $2\leqslant k_2\leqslant|G|+1$. Обозначим $s=k_1-1$, $p=k_2-k_1$. Заметим, что из ограничений на k_1, k_2 следует, что $0\leqslant s<|G|$, $1\leqslant p\leqslant|G|$ и $s+p\leqslant|G|$. Пара найденных равных элементов в новых обозначениях запишется как $(s+1)h$ и $(s+p+1)h$. Из детерминированности групповой операции следует, что бесконечная цепочка $h, 2h, 3h, \ldots$ образует периодическую с периодом p последовательность, начиная с элемента $(s+1)h$, откуда следует утверждение леммы.

Теперь отметим, что в одномерном случае существуют только 3 локатора (полный, смотрящий влево и смотрящий вправо), но значение полного локатора всегда равно сумме значений левого и правого локаторов. Поэтому всегда можно считать, что в одномерном случае имеются ровно два локатора: левый и правый.

Если $X, Y\in\mathbb{Z}$, $T\in\mathbb{N}$, $X<Y$, $T<Y-X$, $[X,Y]=\{x\in\mathbb{Z}\colon X\leqslant x\leqslant Y\}$, то состояние (g,f) одномерного клеточного автомата будем называть *периодичным с периодом T на отрезке* $[X,Y]$, если функции f и g имеют (не обязательно минимальный) период T на этом отрезке, т. е. если для любого $x\in[X,Y-T]$ выполнено $g(x+T)=g(x)$ и $f(x+T)=f(x)$.

Л е м м а 2. *Пусть $I=(g,f)$ — периодичное состояние с периодом T на отрезке $[X,Y]$, $n=Y-X+1$, $B=\sum\limits_{x=X}^{X+T-1}g(x)$ — элемент полугруппы G, p и s — соответственно период и предпериод элемента B, $n>(2(s+1)+p)T$. Тогда любой одномерный клеточный автомат с локаторами $\sigma=(\mathbb{Z},Q,V,G,+,L,\varphi,\psi)$ с начальным состоянием I в следующий такт перейдет в состояние $I_1=\Phi_\sigma(g,f)$, которое будет периодичным с периодом pT на отрезке $[X+(s+1)T; Y-(s+1)T]$.*

Чтобы показать периодичность $I_1=\Phi_\sigma(g,f)$ с периодом pT на отрезке $[X+(s+1)T; Y-(s+1)T]$, достаточно показать, что состояния и значения левого и правого локаторов периодичны на этом отрезке с периодом pT. Периодичность состояний очевидна, поскольку по условию они периодичны с периодом T.

Рассмотрим любую ячейку x из отрезка $[X+(s+1)T; Y-(s+1)T]$ и ячейку $x+pT$. Значение левого локатора ячейки $x+pT$ на pB больше значения левого локатора ячейки x, но, поскольку p — период элемента B, то эти два значения равны. Аналогичные рассуждения можно провести и для значений правого локатора. Тем самым мы показали периодичность с периодом pT значений левого и правого локаторов, а значит, и периодичность I_1 с периодом pT.

Введем операции удлинения периодической части. Пусть (g,f) — периодичное состояние с периодом T на отрезке $[X,Y]$, $Y-X+1>T$. Тогда *операция удлинения периодической части справа* U_T^m добавляет к периодической части справа m фрагментов длины T так, чтобы полученное состояние было периодичным на $[X,Y+mT]$ и чтобы хвост исходного состояния после Y и хвост удлиненного состояния после $Y+mT$ совпадали, т. е. $U_T^m(g,f)=(g',f')$ такое преобразование, что если $x\leqslant Y$, то $g'(x)=g(x)$, $f'(x)=f(x)$, если $Y<x\leqslant Y+mT$, то $g'(x)=g'(x-T)$, $f'(x)=f'(x-T)$ и если $x>Y+mT$, то $g'(x)=g(x-mT)$, $f'(x)=f(x-mT)$. Аналогично *операция удлинения периодической части слева* $W_T^m(g,f)=(g',f')$ такое преобразование, что если $x\geqslant X$, то $g'(x)=g(x)$, $f'(x)=f(x)$, если $X-mT\leqslant x<X$,

то $g'(x)=g'(x+T)$, $f'(x)=f'(x+T)$ и если $x<X-mT$, то $g'(x)=g(x+mT)$, $f'(x)=f(x+mT)$.

Для операций удлинения периодической части справа и слева справедливо следующее утверждение.

Лемма 3. *Пусть* $T,m\in\mathbb{N}$, $X,Y\in\mathbb{Z}$, $X<Y$, (g,f) *— периодичное состояние с периодом* T *на отрезке* $[X,Y]$, $n=Y-X+1$, $n>(2(s+1)+p)T$, $B=\sum\limits_{x=X}^{X+T-1} g(x)$ *— элемент полугруппы* G, p *и* s *— соответственно период и предпериод элемента* B, $(u,v)=U_T^{pm}(g,f)$, $(w,z)=W_T^{pm}(g,f)$, $\sigma=(\mathbb{Z},Q,V,G,+,L,\varphi,\psi)$ *— одномерный клеточный автомат с локаторами,* $(g',f')=\Phi_\sigma(g,f)$, $(u',v')=\Phi_\sigma(u,v)$ *и* $(w',z')=\Phi_\sigma(w,z)$. *Тогда* (g',f') *— периодичное с периодом* pT *на отрезке* $[X+(s+1)T,Y-(s+1)T]$ *и* $(u',v')=U_{pT}^m(g',f')$, $(w',z')=W_{pT}^m(g',f')$.

Смысл данной леммы заключается в том, что если в середину периодической части добавить несколько периодов, то состояния и сигналы вещания на хвостах (что на левом, что на правом) на следующий такт не поменяются по сравнению с теми, что были бы, если бы эти периоды не добавлять. Доказывается этот факт аналогично доказательству предыдущей леммы с использованием того, что p — период сигнала вещания B.

Лемма 4. *Пусть* $T,m,t\in\mathbb{N}$, $X,Y\in\mathbb{Z}$, $X<Y$, $n=Y-X+1$, $a=\lfloor\frac{X+Y}{2}\rfloor$, (g_0,f_0) *— периодичное состояние с периодом* T *на отрезке* $[X,Y]$, $\sigma=(\mathbb{Z},Q,V,G,+,L,\varphi,\psi)$ *— одномерный клеточный автомат с локаторами,* $M=|G|>1$, $n>2MT\frac{M^t-1}{M-1}+2TM^t$, $(u_0,v_0)=U_T^{m(M!)^t}(g_0,f_0)$, $(w_0,z_0)=W_T^{m(M!)^t}(g_0,f_0)$, $(g_0,f_0),(g_1,f_1),(g_2,f_2),\ldots$, $(u_0,v_0),(u_1,v_1),(u_2,v_2),\ldots$, $(w_0,z_0),(w_1,z_1),(w_2,z_2),\ldots$ *— поведения клеточного автомата с локаторами* σ *для начальных состояний* (g_0,f_0), (u_0,v_0), (w_0,z_0) *соответственно, т. е.* $(g_{i+1},f_{i+1})=\Phi_\sigma(g_i,f_i)$, $(u_{i+1},v_{i+1})=\Phi_\sigma(u_i,v_i)$, $(w_{i+1},z_{i+1})=\Phi_\sigma(w_i,z_i)$, *для всех* $i=0,1,2,\ldots$. *Тогда для любого целого* i, $0\leqslant i\leqslant t$, *выполнено* $g_i(a)=u_i(a)=w_i(a)$ *и* $f_i(a)=v_i(a)=z_i(a)$.

Если обозначить $X_i=X+MT\frac{M^i-1}{M-1}$, $Y_i=Y-MT\frac{M^i-1}{M-1}$, $i=0,1,\ldots,t$, то индукцией по i, $i=0,1,\ldots,t$, можно показать, что (g_i,f_i) — периодичное с периодом p_i, $p_i\leqslant M^iT$, на отрезке $[X_i,Y_i]$, $(u_i,v_i)=U_{p_i}^{mr_i(M!)^{t-i}}(g_i,f_i)$ и $(w_i,z_i)=W_{p_i}^{mr_i(M!)^{t-i}}(g_i,f_i)$, где p_i,r_i — некоторые натуральные числа.

Базис индукции следует из условий леммы, а индуктивный переход доказывается с помощью леммы 3.

Остается заметить, что число $a=\lfloor\frac{X+Y}{2}\rfloor$ является серединой каждого из отрезков $[X_i,Y_i]$, $i=0,1,\ldots,t$. Поэтому из определения операций удлинения периодической части справа и слева сразу получаем, что $g_i(a)=u_i(a)=w_i(a)$ для любого i, $0\leqslant i\leqslant t$.

Теперь, используя лемму 4, можем доказать теорему 4.

Рассмотрим произвольное общее положение задачи поиска ближайшего соседа $I_0=(g_0,f_0)$. Пусть s_L — расстояние от «центральной» ячейки до ближайшего левого соседа, а s_R — до ближайшего правого соседа. Пусть $s=\min(s_L,s_R)$.

Предположим, что некоторый одномерный клеточный автомат с локаторами решает задачу за время t, и $t\leqslant\log_M(\frac{s-1}{6})$, где M — мощность

алфавита вещания G. Отсюда получаем $s-1 \geqslant 6M^t > 2M\frac{M^t-1}{M-1}+2M^t$. Отметим, что в начальной конфигурации между «центральной» ячейкой и ближайшим соседом ровно $s-1$ ячейка в состоянии покоя и каждая из этих ячеек посылает в эфир нейтральный элемент группы. Следовательно, начальное состояние является периодичным с периодом 1 на отрезке длины $s-1$ между «центральной» ячейкой и ближайшим соседом. Пусть ячейка a является серединой этого отрезка.

Для определенности предположим, что $s_L > s_R$. В финальном состоянии, т. е. в момент t ячейка a по определению окажется в состоянии q_{RF}.

Возьмем натуральное m такое, что $m(M!)^t > s_L - s_R$. Рассмотрим начальное состояние $(u_0, v_0) = U_1^{m(M!)^t}(g_0, f_0)$, являющееся удлинением периодической части справа. Поскольку $m(M!)^t > s_L - s_R$, то в этом состоянии ближайшим соседом окажется ячейка, находящаяся слева от «центральной» на расстоянии s_L. Поскольку начальные состояния (g_0, f_0) и (u_0, v_0) удовлетворяют условиям леммы 4 и поскольку $s-1 > 2M\frac{M^t-1}{M-1}+2M^t$, то, согласно лемме 4, в момент t будет выполнено $v_t(a) = f_t(a) = q_{RF}$. Но этого не может быть, поскольку для начального состояния (u_0, v_0) ближайшая ячейка находится слева.

Получили противоречие. Следовательно, предположение, что существует одномерный клеточный автомат с локаторами, который решает задачу за время $t \leqslant \log_M(\frac{s-1}{6})$, было неверным.

Случай, когда $s_L \leqslant s_R$, рассматривается аналогично, только надо будет удлинять периодическую часть влево.

Теорема 4 доказана.

Можно заметить, что теорема 4 фактически показывает, что задача сравнения длин отрезков не может быть решена с помощью одномерного клеточного автомата с локаторами быстрее по порядку, чем логарифм от длины меньшего отрезка.

Также заметим, что, учитывая описанный выше алгоритм решения задачи поиска ближайшего соседа, мы получили, что время решения этой задачи одномерным клеточным автоматом с локаторами по порядку равно логарифму от расстояния до ближайшего соседа.

Самое интересное, что, казалось бы, более сложная задача — двумерная задача поиска ближайшего соседа — решается за константное время. Для этого надо использовать клеточный автомат с 9 локаторами, один из которых полный, а остальные образуют лучи с шагом 45 градусов. Этот результат доказан Васильевым в работе [3].

Опишем приведенный в [3] двумерный клеточный автомат с локаторами и соответствующий ему алгоритм решения двумерной задачи поиска ближайшего соседа.

Уточним формулировку задачи поиска ближайшего соседа для двумерного случая.

Пусть на плоскости $\mathbb{Z}^2$ задано начальное состояние I клеточного автомата, удовлетворяющее следующим условиям:

— любой ячейке присвоено одно из трех состояний $\{q_S; q_C, *\}$;

— есть лишь одна ячейка, которой присвоено состояние q_C;

— есть лишь конечное и непустое множество ячеек, которым присвоено состояние q_S.

Решением задачи поиска ближайшего соседа, соответствующей начальному состоянию I, назовем состояние автомата, удовлетворяющее следующим условиям:

— ячейке, которой в I было присвоено состояние q_C, присвоено состояние q_{CF};

— ячейке, ближайшей к ячейке в состоянии q_{CF}, из тех, которым в I было присвоено состояние q_S, присвоено состояние q_{SF}. Если таких ячеек несколько, то одной (произвольной) из них присваивается состояние q_{SF}, а остальным — $*$;

— остальные ячейки находятся в состоянии $*$.

Определим, что клеточный автомат с локаторами σ решает задачу поиска ближайшего соседа, если его начальное состояние удовлетворяет условиям, описанным выше, и его финальное состояние существует и соответствует решению задачи поиска ближайшего соседа для его начального состояния. При этом такой автомат, перейдя в состояние, соответствующее решению какой-либо задачи поиска ближайшего соседа, должен в нем оставаться во всех дальнейших тактах.

Теорема 5 (Д. И. Васильев [3]). *Существует клеточный автомат σ с* 15 *состояниями и с мощностью алфавита вещания* 40, *который решает двумерную задачу поиска ближайшего соседа за время, не превосходящее* 13.

Рассмотрим клеточный автомат с локаторами σ с набором локаторов, состоящем из локаторов, изображенных на рис. 4, и полного локатора. Полугрупповой операцией на алфавите вещания будет максимум.

Зададим на клеточном пространстве систему координат с центром в центральной ячейке. Центральная ячейка в построенном автомате постоянно отправляет в эфир сигнал 1. Ячейки, получившие такой сигнал с локатора $R3$, поймут, что находятся на верхней координатной полуоси. Аналогичным образом каждая ячейка может идентифицировать свое нахождение на остальных трех координатных полуосях. Ячейки, находящиеся на полуосях, постоянно отправляют в эфир сигнал с номером своей полуоси (верхняя — 2, правая — 3, нижняя — 4 и левая — 5). По этим сигналам каждая ячейка может распознать, в какой координатной четверти она находится. Например, получив сигнал 2 с локатора $R4$ и сигнал 3 с локатора $R3$, можно

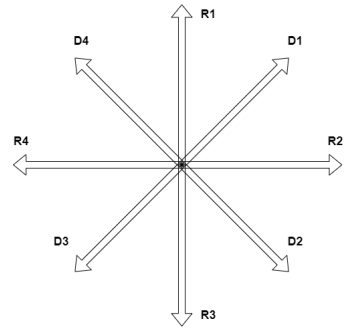

Рис. 4. Направления и названия локаторов

однозначно определить, что рассматриваемая ячейка находится в первой координатной четверти. Идея функционирования построенного автомата состоит в том, чтобы спроецировать вдоль манхэттенской окружности все точки из задачи на одну полуось, найти ближайшую к центру проекцию, а затем восстановить ее прообраз. Например, точка из первой четверти может послать специальный сигнал, считываемый только правой полуосью. Точка из правой полуоси, получив такой сигнал с локатора $D4$, поймет, что является проекцией одной из точек задачи (рис. 5, слева). Повторив такую итерацию 4 раза, можно спроецировать все точки на верхнюю полуось (рис. 5, справа).

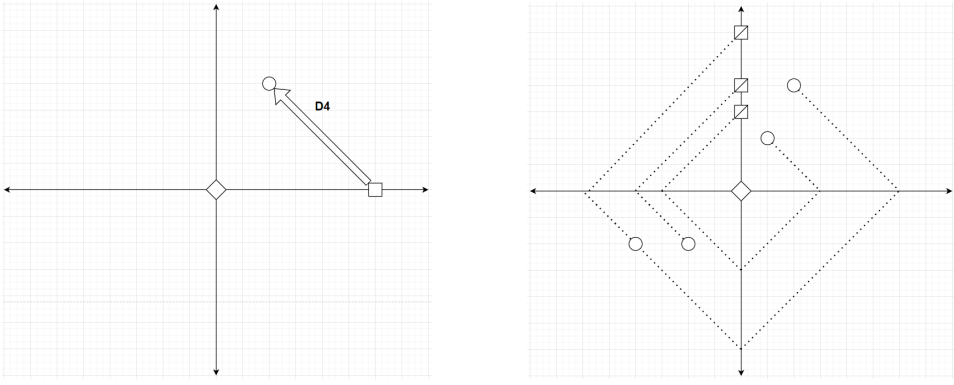

Рис. 5. Пример проецирования одной точки на полуось (слева), ход проецирования задачи на верхнюю полуось (справа)

Чтобы найти ближайшего соседа на верхней полуоси, каждому кандидату достаточно послать в эфир специальный сигнал и, получив такой сигнал с локатора $R3$, самоустраниться (т. е. перейти в состояние покоя). После того, как ближайшая проекция найдена, достаточно обратным ходом описанного алгоритма восстановить ее прообраз.

§ 11. Сложение векторов

Ранее мы сформулировали задачу сравнения векторов на прямой и привели ее решение с помощью клеточных автоматов и показали, что время решения в этом случае пропорционально длинам векторов.

В работе [10] Д. Э. Ибрагимова показала, что с помощью одномерного клеточного автомата с локаторами эту задачу можно решить за время, приблизительно равное удвоенному логарифму от длины меньшего вектора.

Идея алгоритма состоит в том, чтобы вычислять двоичное представление меньшего вектора, аналогично тому, как это делалось в задаче поиска ближайшего соседа (алгоритм Васильева), и далее постепенно откладывать этот вектор из конца большего вектора. Когда векторы смотрят в одну сторону, легко понять какой из векторов короче, а когда они смотрят в разные стороны, Ибрагимова предлагает сравнить длины векторов по алгоритму Васильева. Отсюда появляются два логарифма от длины меньшего вектора.

Если же не сравнивать длины векторов в случае, когда векторы смотрят в разные стороны, а откладывать, например, правый вектор из конца левого,

то мы получим один логарифм, но уже от длины большего вектора (если правый вектор будет больше левого).

Опишем более подробно последний алгоритм.

В начальной конфигурации одна клетка находится в состоянии B (начало координат) и одна или две клетки находятся в состоянии E (концы векторов). Когда имеется только одна клетка в состоянии E, то это будем понимать как то, что у обоих векторов концы совпали. Остальные клетки находятся в состоянии покоя $*$.

Напомним, что клеточный автомат не может «понять», одна или две клетки находятся в состоянии E, тогда как клеточный автомат с локаторами это легко понимает.

Например, начальное состояние может быть следующим.

Таблица 5

$t = 0$

f:	*	B	*	*	*	*	*	E	*	E	*	*	*	*	*	*	*	*

Один из векторов назовем *закрепленным*, а второй *перемещаемым*. Перемещаемый вектор мы потом будем перемещать к концу закрепленного вектора.

Алгоритм решения будет состоять из нескольких этапов.

На первом этапе (этапе ориентации) мы определим, по какую сторону от начала координат находятся концы векторов. Это будем делать также, как и в алгоритме Васильева. Далее назначим закрепленный и перемещаемые векторы. А именно если векторы смотрят в одну сторону, то закрепленным сделаем более длинный вектор, а перемещаемым сделаем более короткий (сравнить длины векторов, направленных в одну сторону, легко, это делается так же, как мы определяли самую левую и самую правую клетки). Если же векторы смотрят в разные стороны, то закрепленным объявим левый вектор, а перемещаемым — правый. В данном алгоритме состояние клетки будет двухкомпонентным. Первую компоненту будем использовать, в частности, чтобы отмечать перемещаемый вектор, а вторую — чтобы откладывать его от конца закрепленного. Все клетки перемещаемого вектора переводим в состояние P. Все клетки луча, начинающегося в конце закрепленного вектора и направленного в ту же сторону, что и перемещаемый вектор, отметим состояниями Z, а начало этого луча (т. е. конец закрепленного вектора) отметим состоянием HZ. Остальные клетки переведем в состояние покоя. На этом первый этап завершается. Понятно, что он займет константное число тактов (обозначим это число через t_0).

После первого этапа могут возникнуть 3 принципиально разные ситуации: оба вектора направлены вправо, оба вектора направлены влево и векторы направлены в разные стороны. Далее мы будем рассматривать только первую ситуацию, а остальные две читатель может разобрать самостоятельно по аналогии с приведенным ниже алгоритмом.

На втором этапе мы будем складывать векторы, а точнее, будем перемещать перемещаемый вектор к концу закрепленного. Делать это будем следующим образом. Сигнал вещания у нас будет четырехкомпонентным. Первые две компоненты будут суммирующими, и алфавит этих компонент

будет $\{0, 1\}$, а полугрупповая операция будет сложение по модулю 2. Вторые две компоненты будут вспомогательными, и полугрупповая операция для этих компонент будет максимум. Первую суммирующую компоненту мы будем использовать для вычисления двоичного представления длины перемещаемого вектора, а вторую суммирующую компоненту — для откладывания перемещаемого вектора от конца закрепленного.

Если обозначить через L_1, L_2, L_3, L_4 значения компонент левого локатора, то по результатам первого этапа состояние, сигналы вещания и значения локаторов будут следующими.

Таблица 6

$t = t_0$

f_1:	$*$	B_P	P	P	P	P	P	P	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$
f_2:	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	HZ	Z	Z	Z	Z	Z	Z	Z	Z
g_1:	0	0	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
L_1:	0	0	0	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0
g_2:	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
L_2:	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	1
g_3:	0	0	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
L_3:	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Если перемещаемый вектор смотрит вправо (наш рассматриваемый случай), то значения первой и второй компонент правого локатора использоваться не будут, поскольку они будут неопределенны, как сумма бесконечного числа единиц.

Состояния и сигналы вещания ячеек для следующего такта будут меняться по следующим правилам:

— если первая компонента состояния равна P и значение первой компоненты левого локатора равно 1, то первая компонента состояния не меняется, а сигналы вещания по первой и третьей компоненте будут равны 1;

— если первая компонента состояния равна P и значение первой компоненты левого локатора равно 0, то первая компонента состояния меняется на $*$, а сигналы вещания по первой и третьей компоненте будут равны 0;

— если вторая компонента состояния равна HZ и значение первой компоненты левого локатора равно 0, то вторая компонента состояния не меняется, а сигнал вещания по второй компоненте будет равен 1;

— если вторая компонента состояния равна HZ и значение первой компоненты левого локатора равно 1, то вторая компонента состояния меняется H, а сигнал вещания по второй компоненте будет равен 0;

— если вторая компонента состояния равна H и значение первой компоненты левого локатора равно 0, а значение третьей компоненты левого локатора равно 1, то вторая компонента состояния не меняется, а сигнал вещания по второй компоненте будет равен 0;

— если вторая компонента состояния равна H и значение третьей компоненты левого локатора равно 0 (завершили получать биты двоичного представления перемещаемого вектора), то вторая компонента состояния меняется на $*$, а сигнал вещания по второй компоненте будет равен 0;

— если вторая компонента состояния равна H и значение первой компоненты левого локатора равно 1, то вторая компонента состояния не меняется, а сигнал вещания по второй компоненте будет равен 1;

— если вторая компонента состояния равна Z и значение второй компоненты левого локатора равно 1 и значение третьей компоненты левого локатора равно 1, то вторая компонента состояния не меняется, а сигнал вещания по второй компоненте будет равен 1;

— если вторая компонента состояния равна Z и значение второй компоненты левого локатора равно 0 и значение третьей компоненты левого локатора равно 1, то вторая компонента состояния меняется на $*$, а сигнал вещания по второй компоненте будет равен 0;

— если вторая компонента состояния равна Z и значение третьей компоненты левого локатора равно 0, то вторая компонента состояния меняется на L, а сигнал вещания по четвертой компоненте будет равен 1;

— если вторая компонента состояния равна L и значение четвертой компоненты левого локатора равно 0 (самая левая ячейка в состоянии L), то вторая компонента состояния меняется на $*$, а первая компонента состояния становится E_F;

— если вторая компонента состояния равна L и значение четвертой компоненты левого локатора равно 1 (не самая левая ячейка в состоянии L), то вторая компонента состояния меняется на $*$;

— если первая компонента состояния равна B_P и значение четвертой компоненты правого локатора равно 1, то первая компонента состояния меняется на B_F;

— во всех не рассмотренных выше случаях компоненты состояние будут равны $*$, а компоненты сигнала вещания будут равны 0.

Приведем состояния для нашего примера в последующие такты.

Здесь состояниями HZ и H отмечается конец закрепленного вектора. На первой компоненте левого локатора ячейки с состояниями HZ или H отображаются очередные биты двоичного представления длины перемещаемого вектора. Состояниями Z отмечаются ячейки, расстояние до которых от конца закрепленного вектора имеет такие же биты в двоичном представлении, как и биты двоичного представления длины перемещаемого вектора. Появление сигнала 0 в третьей компоненте левого локатора (в нашем примере это момент $t = t_0 + 3$) означает, что все биты двоичного представления длины перемещаемого вектора уже получены и пора фиксировать конец вектора суммы. Для этого в момент $t = t_0 + 3$ все оставшиеся ячейки в состоянии Z переходят в состояние L и посылают в эфир по 4 компоненте сигнал 1, а в момент $t = t_0 + 4$ с помощью этих сигналов выбирают самую

Таблица 7

$t = t_0 + 1$

f_1 :	$*$	B_P	$*$	P	$*$	P	$*$	P	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$
f_2 :	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	HZ	$*$	Z	$*$	Z	$*$	Z	$*$	Z
g_1 :	0	0	0	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0
L_1 :	0	0	0	0	1	1	0	0	1	1	1	1	1	1	1	1	1	1
g_2 :	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	1	0	1
L_2 :	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	1	0	0
g_3 :	0	0	0	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0
L_3 :	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Таблица 8

$t = t_0 + 2$

f_1 :	$*$	B_P	$*$	$*$	$*$	P	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$
f_2 :	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	H	$*$	Z	$*$	$*$	$*$	Z	$*$	$*$
g_1 :	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
L_1 :	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1
g_2 :	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0
L_2 :	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0
g_3 :	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
L_3 :	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1

Таблица 9

$t = t_0 + 3$

f_1 :	$*$	B_P	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$
f_2 :	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	H	$*$	$*$	$*$	$*$	$*$	Z	$*$	$*$
g_1 :	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
L_1 :	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
g_2 :	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0
L_2 :	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0
g_3 :	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
L_3 :	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

левую ячейку в состоянии L и эта ячейка в момент $t = t_0 + 5$ становится концом вектора суммы. Задача решена.

Тем самым мы показали, что задачу сложения векторов на прямой можно решить за время, равное логарифму от длины перемещаемого вектора плюс некая небольшая константа.

Теперь покажем, что эту задачу нельзя решить по порядку быстрее, чем за логарифмическое время.

Теорема 6. *Существует такая константа c, что для любого одномерного клеточного автомата с локаторами σ время решения за-*

Таблица 10

$t = t_0 + 4$

f_1 :	$*$	B_P	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$
f_2 :	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	L	$*$	$*$
g_1 :	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
L_1 :	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
g_2 :	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
L_2 :	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
g_3 :	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
L_3 :	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
g_4 :	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
L_4 :	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Таблица 11

$t = t_0 + 5$

f_1 :	$*$	B_F	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	E_F	$*$	$*$
f_2 :	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$
g_1 :	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
L_1 :	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
g_2 :	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
L_2 :	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
g_3 :	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
L_3 :	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
g_4 :	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
L_4 :	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

дачи сложения векторов на прямой будет не меньше, чем $c \log n$, *где* n — *длина более короткого вектора*.

Доказательство. Предположим обратное, что существует одномерный клеточный автомата с локаторами σ, для которого время решения задачи сложения векторов на прямой будет равно $T_\sigma(n) = o(\log n)$.

Рассмотрим начальную конфигурацию, в которой векторы смотрят в разные стороны и длина более короткого вектора равна n. Согласно предположению, мы ее можем решить за время $T_\sigma(n) = o(\log n)$. Но из решения задачи сложения векторов мы сразу (плюс пару тактов) можем решить задачу сравнения длин или задачу поиска ближайшего соседа. Но эти задачи не могут быть решены быстрее, чем за $c \log n$, где c — некоторая константа. Получили противоречие. Теорема доказана.

Интересно то, что если перейти от одномерного клеточного автомата с локаторами к двумерному, то задачу сложения векторов можно решить за константное время.

Задачу сложения векторов на прямой назовем задачей в общем положении, если в начальной конфигурации есть одна ячейка в состоянии B и две ячейки в состоянии E.

Т е о р е м а 7. *Существует двумерный клеточный автомат с локаторами с 7 сигналами вещания и 17 состояниями, который решает любую задачу сложения векторов на прямой в общем положении за 5 тактов.*

Д о к а з а т е л ь с т в о. Рассмотрим следующий двумерный клеточный автомат с локаторами $\sigma = (\mathbb{Z}^2, Q, \varnothing, E_7, \max, L, \varphi, \psi)$, где $Q = \{*, B, E, B^1, E^1, B^2, L^2, R^2, B^3, L^3_L, L^3_R, R^3, B^4, L^4_L, L^4_R, B_F, E_F\}$, и где $*$ — состояние покоя, шаблон соседства пустой, сигналы вещания образуют полугруппу $(E_7, \max)$, шаблон локаторов L состоит из 8 локаторов, изображенных на рис. 4; функция переходов φ и функция вещания ψ будут описаны ниже.

Идея алгоритма решения состоит в том, чтобы отобразить конец левого вектора на ось ординат, затем перенести его на прямую, параллельную оси ординат, но проходящую через конец правого вектора, а затем полученную точку обратно отобразить на ось абсцисс.

Клеточный автомат с локаторами σ будет работать ровно 5 тактов, и мы будем описывать значения функции переходов и функции вещания на каждом такте, при этом демонстрируя их на трех ситуациях, когда в начальной конфигурации оба вектора направлены вправо, векторы направлены в разные стороны, оба вектора направленны влево. Эти начальные состояния и дальнейшее поведение для этих начальных конфигураций изображены на рис. 6, 7, 8 соответственно. Отметим также, что на этих рисунках ячейки, которые посылают сигналы в эфир, обведены кружком, а рядом с кружком указано значение сигнала вещания, посылаемого в эфир. Ячейки без кружков посылают в эфир нейтральный элемент 0. Состояние покоя на рисунках изображено в виде пустой клетки.

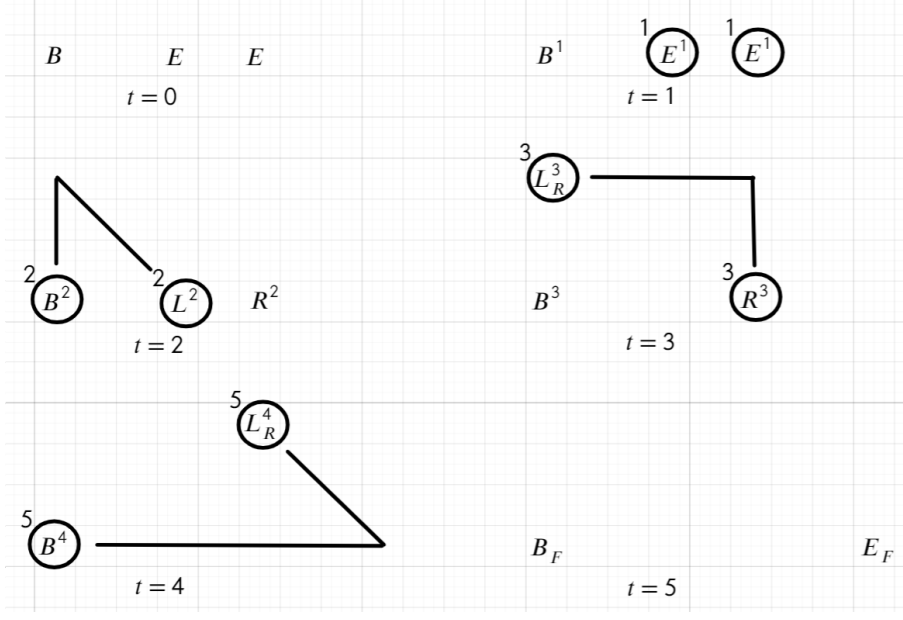

Рис. 6. Оба вектора направлены вправо

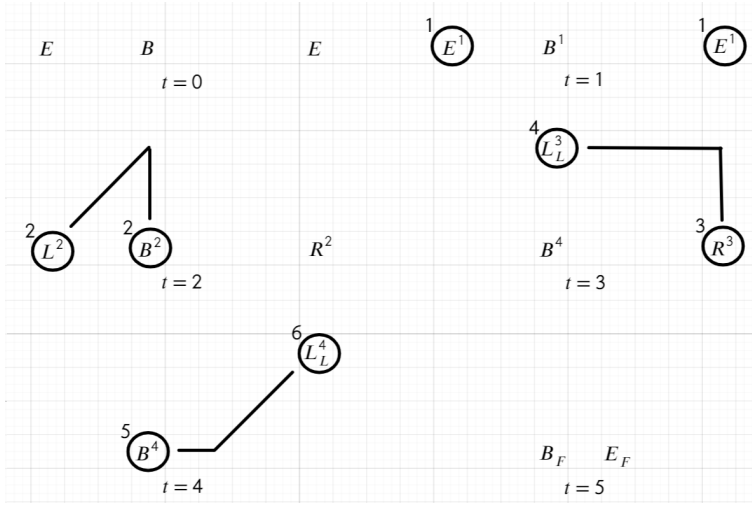

Рис. 7. Векторы направлены в разные стороны

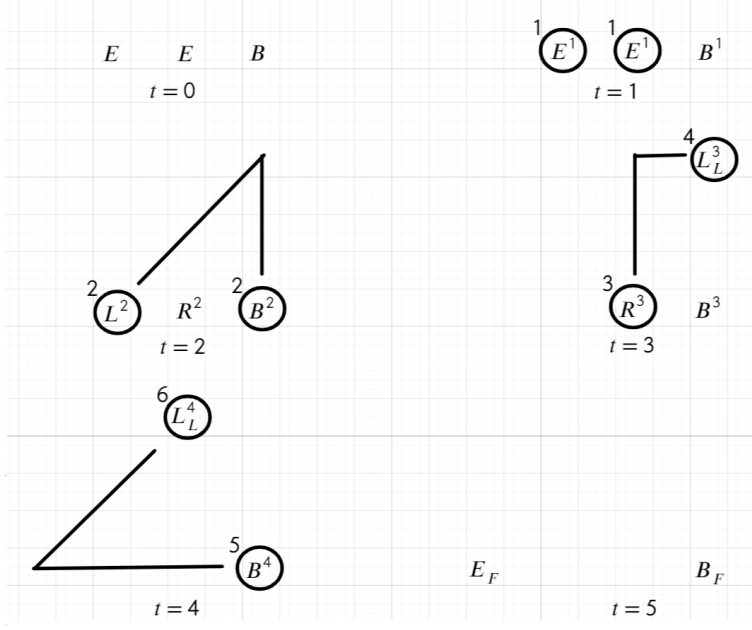

Рис. 8. Оба вектора направлены влево

Для упрощения описания мы будем говорить, что ячейка на таком-то такте посылает в эфир такой-то сигнал, но решение о посылаемом сигнале принималось на предыдущем такте.

На нулевом такте ячейка в состоянии B (начало координат) переходит в состояние B^1. Ячейки в состоянии E (концы векторов) принимают решение о посылке в эфир сигнала 1 и переходят в состояние E^1.

На первом такте ячейка в состоянии B^1 переходит в состояние B^2. Ячейки в состояниях E^1 посылают в эфир сигналы 1. Ячейка в состоянии E^1, которая получает сигнал 1 на локатор R4 (левый локатор), понимает, что она правее и переходит в состояние R^2. Ячейка в состоянии E^1, которая получает сигнал 1 на локатор R2 (правый локатор), понимает, что она левее и переходит в состояние L^2.

На втором такте ячейка в состоянии B^2 посылает в эфир сигнал 2 и переходит в состояние B^3. Ячейка в состоянии R^2 переходит в состояние R^3. Ячейка в состоянии L^2 посылает в эфир сигнал 2 переходит в состояние покоя. Ячейка в состоянии покоя, которая получает сигналы 2 на локаторы R3 (снизу) и D2 (снизу справа), переходит в состояние L^3_R. Ячейка в состоянии покоя, которая получает сигналы 2 на локаторы R3 (снизу) и D3 (снизу слева), переходит в состояние L^3_L.

На третьем такте ячейка в состоянии B^3 переходит в состояние B^4. Ячейка в состоянии R^3 посылает в эфир сигнал 3 переходит в состояние покоя. Ячейка в состоянии L^3_R посылают в эфир сигнал 3. Ячейка в состоянии L^3_L посылают в эфир сигнал 4. Ячейка в состоянии покоя, которая получает сигналы 3 на локаторы R3 (снизу) и R4 (слева), переходит в состояние L^4_R. Ячейка в состоянии покоя, которая получает сигнал 3 на локатор R3 (снизу) и сигнал 4 на локатор R4 (слева) или на локатор R2 (справа), переходит в состояние L^4_L.

На четвертом такте ячейка в состоянии B^4 посылает в эфир сигнал 5 и переходит в состояние B_F. Ячейка в состоянии L^4_R посылает в эфир сигнал 5 переходит в состояние покоя. Ячейка в состоянии L^4_L посылает в эфир сигнал 6 переходит в состояние покоя. Ячейка в состоянии покоя, которая получает сигналы 5 на локаторы R4 (слева) и D4 (сверху слева), переходит в состояние E_F. Ячейка в состоянии покоя, которая получает сигнал 5 на локаторы R4 (слева) или R2 (справа) и сигнал 6 на локатор D1 (сверху справа), переходит в состояние E_F.

На пятом такте остаются только две активные ячейки в состояниях B_F и E_F или только одна ячейка в состоянии B_F, если векторы были направлены в разные стороны и имели одинаковую длину.

Теорема доказана.

Понятно, что таким же образом можно решать и задачу сложения векторов на прямой не в общем положении, просто надо будет предусмотреть большее число случаев.

Более того, можно за константное время решить задачу сложения векторов на плоскости. Для этого надо отобразить концы векторов на оси координат. Для каждой из осей координат решить задачу сложения векторов на прямой, а потом восстановить конец суммарного вектора.

§ 12. Реализация баз данных типа «ключ—значение»

Приведем еще один пример практически важной задачи — задачи реализации баз данных типа «ключ—значение». Как показали Э. Э. Гасанов и А. А. Пропажин в работе [7], эту задачу удобно решать с помощью клеточных автоматов с локаторами.

База данных «ключ—значение» — это популярная сейчас парадигма хранения данных, также называемая словарем. Такую базу данных можно

представлять в виде множества пар строк (k, v), где первая строка k называется ключом и служит идентификатором пары, а вторая строка v называется значением. *Строка* — это последовательность символов некоторого алфавита A, оканчивающаяся специальным символом 0, называемым *символом окончания строки*, причем символ 0 не принадлежит алфавиту A.

База данных «ключ—значение» поддерживает следующие операции:

1) *вставка пары* (k, v) — в базе данных появляется запись с ключом k и значением v; если запись с ключом k уже имелась в базе данных, то значение заменяется на v;

2) *удаление записи с ключом* k — из базы данных удаляется запись (k, v); если записи с ключом k в базе данных нет, то база данных не изменяется;

3) *поиск элемента по ключу* k — в базе данных находится запись (k, v) и значение v возвращается в качестве ответа; если записи с ключом k в базе данных нет, то ответом служит пустое множество.

В работе [7] утверждается, что существует клеточный автомат с локаторами и пользователем, который реализует базу данных типа «ключ—значение» и для которого время выполнения операций поиска, вставки и удаления не будет превышать суммарную длину ключа и значения. Приведем данный результат более подробно.

Введем еще одну сущность — *пользователя* базы данных. Будем считать, что пользователь базы данных имеет возможность посылать в эфир сигналы из алфавита вещания и получать из эфира сигналы из алфавита вещания. Будем считать, что клеточный автомат с локаторами вместе с пользователем реализуют базу данных типа «ключ—значение», если алфавит вещания представляет собой множество, состоящее из пар вида («команда», $A \cup \{0\}$), где «команда» — принимает одно из значений: «поиск», «вставка», «удаление», «ответ», «нет ответа», «нет команды» (здесь «нет команды» является минимальным элементом), и поведение клеточного автомата задается следующим образом.

1. Пользователь подает в эфир команду «поиск» и первый символ ключа k, а затем последовательно все остальные символы ключа, включая символ 0. Если в базе данных нет записи с ключом k, то на следующий такт после подачи символа 0 клеточный автомат выдает в эфир команду «нет ответа». Если в базе данных есть запись (k, v), то на следующий такт после подачи символа 0 клеточный автомат выдает в эфир пару («ответ», a), где $a \in A$ — первый элемент значения v, и в последующие такты клеточный автомат последовательно выдает в эфир все остальные символы значения v вплоть до символа 0 включительно.

2. Пользователь подает в эфир команду «вставка» и первый символ ключа k, а потом последовательно все остальные символы ключа, включая 0, и после этого таким же способом передаются символы значения v. В результате в базе данных, реализуемой клеточным автоматом, появляется пара (k, v), т. е. если в последующем подать на вход клеточному автомату команду «поиск» и ключ k, то клеточный автомат вернет значение v в ответ.

3. Пользователь подает команду «удаление» и первый символ ключа k, а потом последовательно все остальные символы ключа, включая 0. В результате из базы данных, реализуемой клеточным автоматом, исчезает запись с ключом k, т. е. при последующем поиске по ключу k клеточный автомат вернет «нет ответа».

Теорема 8 (Э. Э. Гасанов, А. А. Пропажин [7]). *Существует клеточный автомат с локаторами и пользователем, который реализует базу данных типа «ключ—значение» и для которого время выполнения операций поиск, вставка, удаление не будут превышать суммарную длину ключа и значения.*

Приведем идею доказательства данной теоремы.

Будем использовать одномерный клеточный автомат с одним полным локатором, причем элементарные автоматы, лежащие в отрицательной области числовой прямой, использоваться не будут. Выше уже был описан алфавит вещания. Алфавит состояний состоит из троек вида («команда», «тип ячейки», $A \cup \{0, *\}$), где «команда» принимает одно из значений: «поиск», «вставка», «удаление», «ответ»; «тип ячейки» принимает одно из значений: «командир», «текущая ячейка типа ключ», «текущая ячейка типа значение», «необрабатываемая ячейка», «приемник для записи»; $*$ — специальный символ. Символом «приемник для записи» будет помечаться элементарный автомат, начиная с которого будет осуществляться следующая запись в базу данных. В начальный момент этим символом будет помечен автомат с номером 0. Записи базы данных будут представлять собой пары строк ключ—значение, причем первый символ каждой записи будет отмечен состоянием «командир».

Опишем функционирование данного автомата. Когда начинается вставка, все командиры получают из эфира сигнал в виде («вставка», k_1), где k_1 — первый символ ключа. Если k_1 совпадает со значением, хранящимся в состоянии клетки командира, то в состоянии следующей справа клетки команда меняется на «вставка», а тип ячейки на «текущая ячейка типа ключ». На следующий такт следующая справа от командира клетка, также получив сигнал из эфира, проверяет совпадение с хранящимся символом. При совпадении происходит аналогичный предыдущему процесс за тем исключением, что в конце такта тип этой клетки изменится на «необрабатываемая ячейка». Процесс происходит последовательно до символа 0 включительно. Процесс проверки ключа начинается одновременно со всех командиров. Если в какой-то момент случилось несовпадение, то тип следующей клетки меняется на «необрабатываемая ячейка». Если дошли до символа 0, то это означает, что мы нашли запись с искомым ключом и эту запись надо исключить из базы данных. Исключения из базы данных достигается тем, что 0 в состоянии текущей клетки меняется на символ $*$. В результате при последующих поисках совпадения не будет.

Одновременно с поиском ключа, начиная с клетки, помеченной состоянием «приемник для записи», происходит последовательная запись пары «ключ—значение» в базу данных. В момент прихода команды «вставка» клетка, находящаяся в состоянии «приемник для записи», становится командиром и сохраняет в своем состоянии первый символ ключа k_1. При этом в следующий момент клетка, находящаяся правее данной клетки, меняет команду на «вставка» и тип ячейки на «приемник для записи». Далее клетки, имеющие команду «вставка» и тип ячейки «приемник для записи», сохраняют в своем состоянии поступающие из эфира символы ключа и значения. При этом состояние («вставка», «приемник для записи») переходит к следующей справа клетке. Исключением является момент прихода символа 0 для значения. В этот момент следующая справа клетка переходит в состояние («поиск», «приемник для записи»).

Удаление происходит аналогично вставке. Только во время удаления не начинается процесс записи в конец базы данных.

Во время поиска считывание ключа происходит так же, как и при вставке. Но, дойдя до символа 0, замена данного символа на $*$ не происходит. Клетка, следующая справа от клетки с символом 0, меняет команду на «поиск» и тип ячейки на «текущая обрабатывающая ячейка типа значение». Клетки, находящиеся в этом состоянии, отправляют в эфир сигнал («ответ», a), где $a \in A$ — хранимый символ. Текущая клетка после отправки сигнала меняет тип на «необрабатываемая ячейка». Сигналы с символами значения последовательно отправляются в эфир до сигнала с символом 0 включительно.

§ 13. Вычислительные устройства на базе клеточных автоматов с локаторами

Хотя при описании клеточного автомата с локаторами мы используем понятие эфира и тем самым как бы предполагаем некую беспроводную связь между элементарными автоматами, эту «беспроводную» связь можно реализовать с помощью обычных проводников и стандартных логических элементов.

Так, в работе [11] Г. В. Калачев показал, как можно реализовать клеточные автоматы с локаторами в виде чипов, если все локаторы являются подпространствами с одной выколотой точкой, а алфавит вещания — идемпотентный моноид.

Калачев предлагает следующим образом представлять реализацию клеточного автомата с локаторами в виде чипа.

Предположим для начала, что алфавит вещания есть полугруппа $(\{0,1\}, \max)$, A — некоторое подпространство пространства $\mathbb{Z}^k$ ячеек клеточного автомата (например, это может быть некоторая прямая), задающее некоторый локатор ν. В качестве эфира должно быть некоторое устройство, «суммирующее» неограниченное количество электрических сигналов из A. В качестве такого устройства может выступать проводник, подключенный ко всем выходам элементов, которые нужно суммировать, и подключенный к усилителю, выход которого подключен ко входам-локаторам всех элементов. Таким образом, если один из элементов послал в эфир сигнал, этот сигнал усилится и на локаторы всех элементов из A придет сигнал 1. Если же все элементы выдали 0, то и из эфира придет 0. Таким образом можно реализовать операцию max от неограниченного числа аргументов, принимающих значения из множества $\{0,1\}$.

Однако для клеточного автомата с локатором ν в состоянии (g, f) требуется уметь для каждой ячейки $\alpha \in A$ вычислять функцию

$$s_\nu(\alpha) = M_1(g, A, \alpha) = \max_{\beta \in A \setminus \{\alpha\}} g(\beta).$$

Можно заметить, что

$$M_1(g, A, \alpha) = \min\Big(\sum_{\beta \in A \setminus \{\alpha\}} g(\beta), 1\Big) = \min\Big(\min\Big(\sum_{\beta \in A} g(\beta), 2\Big) - g(\alpha), 1\Big).$$

Операцию $M_2(A) = \min\bigl(\sum\limits_{\beta\in A} g(\beta), 2\bigr)$ также возможно реализовать, но сложнее, чем операцию max. Например, это можно сделать следующим образом. Каждый вход, представляющий аргумент операции, равный 1, выдает ограниченный ток на провод, соединяющий все аргументы и подключенный к нулевому проводу через резистор. В зависимости от числа входов, равных 1, на соединяющем проводнике будет различное напряжение. Сам проводник можно подключить к двум компараторам, из которых один срабатывает при напряжении, когда хотя бы один вход активен, а второй срабатывает при напряжении, когда хотя бы 2 входа активны. Используя результаты сравнения с компараторов, легко получить значение функции M_2. Затем по общему проводу можно подвести результат $s = M_2(A)$ обратно ко всем ячейкам из A и ячейке α вычислить $\min(s - g(\alpha), 1)$; получим таким образом в ячейке α требуемый результат $s_\nu(\alpha)$.

Теперь рассмотрим общий случай, когда $(G, +)$ — идемпотентный коммутативный моноид и он задает алфавит вещания, где $G = \{e, e_1, \ldots, e_{n-1}\}$, e — нейтральный элемент G. Пусть, как и ранее, локатор ν задается подпространством $A \subseteq \mathbb{Z}^k$.

Закодируем не нейтральные элементы $e_1, \ldots, e_{n-1}$ наборами $(1, 0, \ldots, 0)$, $(0, 1, 0, \ldots, 0)$, $\ldots$, $(0, \ldots, 0, 1) \in \{0, 1\}^{n-1}$, а нейтральный элемент e закодируем набором из всех нулей. Пусть теперь ячейки клеточного автомата посылают в эфир сигналы $g = (g_1, g_2, \ldots, g_{n-1})$ в соответствии с данной схемой кодирования.

Тогда если мы заведем $n-1$ описанную выше схему M_1, то для каждой ячейки $\alpha \in A$ получим операцию

$$M_3(g, A, \alpha) = (M_1(g_1, A, \alpha), M_1(g_2, A, \alpha), \ldots, M_1(g_{n-1}, A, \alpha)),$$

с помощью которой до каждой ячейки α может быть донесена информация, какие именно элементы полугруппы G встречаются в сумме

$$s_\nu(\alpha) = \sum_{\beta\in A\setminus\{\alpha\}} g(\beta),$$

а с учетом идемпотентности моноида этой информации достаточно, чтобы вычислить $s_\nu(\alpha)$. Тем самым нам остается только вычислить булев оператор:

$$F(x_1, \ldots, x_{n-1}) = \sum_{i=1}^{n-1} x_i \cdot e_i.$$

А этот оператор легко можно реализовать с помощью обычной схемы из функциональных элементов.

Итак, мы получили, что для любого конечного идемпотентного моноида можно реализовать его полугрупповую операцию от неограниченного числа элементов, используя фиксированную СФЭ и несколько проводников, подключенных ко всем элементам, выходы которых суммируются.

С реализацией направленных локаторов (типа лучей или других полупространств) все обстоит хуже. Проводник проводит одинаково во все стороны. Если же использовать диоды, пропускающие ток только в одну сторону, глубина схемы тут же станет линейной по числу аргументов и здесь уже нельзя говорить, что сигнал по эфиру распространяется мгновенно, таким

образом, теряется смысл использования данной модели. Поэтому описанным способом можно реализовать лишь телесные углы, совпадающие с подпространствами. Например, полный локатор Ω реализуется, если соединить пластиной выходы всех ячеек. Можно сделать слой с множеством проводов, идущих в одном направлении, и таким образом будет реализовываться локатор $\{v, -v\}$, где v — направление проводов в данном слое.

Рассмотрим пример реализации клеточного автомата с двумя локаторами в виде вертикальных и горизонтальных прямых. На рис. 9 изображен первый слой интегральной схемы, на котором реализуется логика ячеек. На рис. 10 изображен второй слой интегральной схемы, на котором проводятся локальные соединения между ячейками. На рис. 11 изображен третий слой интегральной схемы, на котором реализуются вертикальные локаторы. На рис. 12 изображен четвертый слой интегральной схемы, на котором реализуются горизонтальные локаторы.

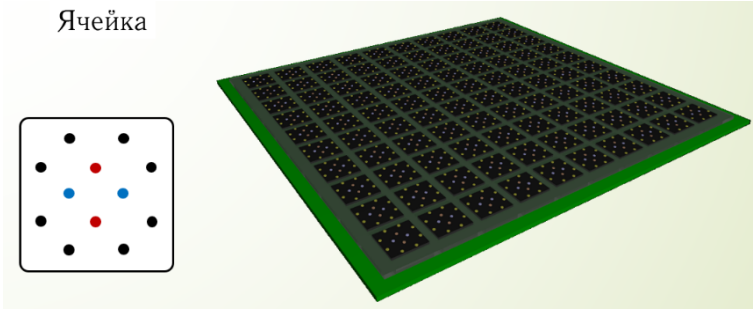

Рис. 9. Первый слой — реализация ячеек

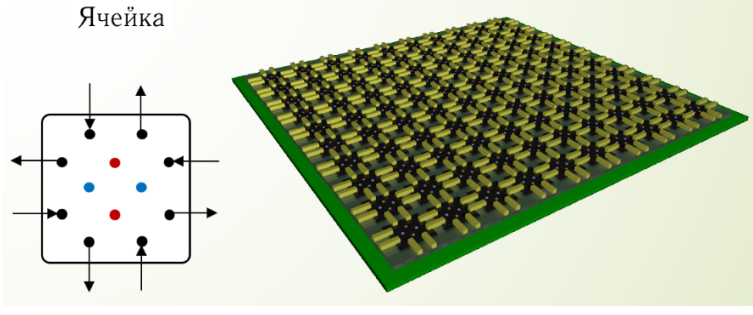

Рис. 10. Второй слой — локальные соединения

Для реализации других (направленных) локаторов требуется использовать какие-то другие физические принципы, выходящие за рамки обычной схемотехники.

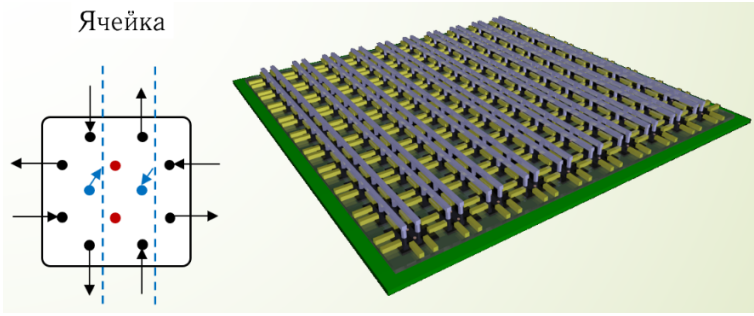

Рис. 11. Третий слой — вертикальные локаторы

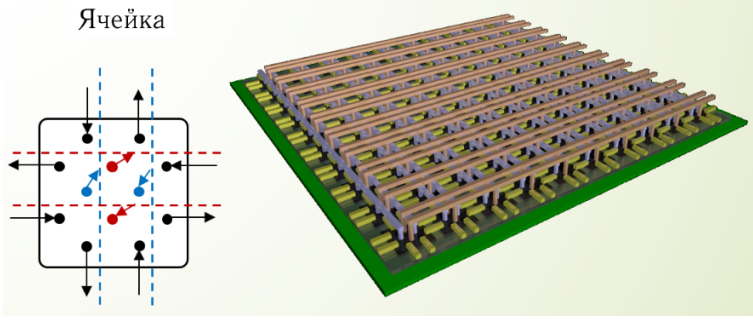

Рис. 12. Четвертый слой — горизонтальные локаторы

СПИСОК ЛИТЕРАТУРЫ

1. Вайнцвайг М. Н. О мощности схем из функциональных элементов // Доклады АН СССР. — 1961. — Т. 159, № 2. — С. 320–323.
2. Васильев Д. И. Поиск ближайшего соседа на прямой с помощью клеточного автомата с локаторами // Интеллектуальные системы. Теория и приложения. — 2020. — Т. 24, № 3. — С. 99–119.
3. Васильев Д. И. Поиск ближайшего соседа на плоскости с помощью клеточного автомата с локаторами // Интеллектуальные системы. Теория и приложения. — 2021. — Т. 25, № 4. — С. 83–87.
4. Васильев Д. И., Гасанов Э. Э. Нижняя оценка сложности задачи поиска ближайшего соседа на прямой с помощью клеточного автомата с локаторами // Вестник МГУ. Серия 1. Математика. Механика. — 2023. — № 5. — С. 33–39.
5. Воротников А. С. Верхние оценки переключательной мощности плоских схем, реализующих автономные автоматные функции // Интеллектуальные системы. Теория и приложения. — 2021. — Т. 25, № 4. — С. 96–99.
6. Гасанов Э. Э. Клеточные автоматы с локаторами // Интеллектуальные системы. Теория и приложения. — 2020. — Т. 24, № 2. — С. 120–133.
7. Гасанов Э. Э., Пропажин. А. А. Реализация баз данных типа «ключ-значение» клеточными автоматами с локаторами // Интеллектуальные системы. Теория и приложения. — 2021. — Т. 25, № 4. — С. 108–112.
8. Гашков С. Б. Глубина булевых функций // Проблемы кибернетики. Вып. 34. — М.: Наука, 1978. — С. 265–268.
9. Гордеева А. С. Моделирование аэродинамики крыла клеточными автоматами // Интеллектуальные системы. Теория и приложения. — 2022. — Т. 26, № 3. — С. 23–46.

10. Ибрагимова Д. Э. Сложение векторов на прямой с помощью клеточного автомата с локаторами // Интеллектуальные системы. Теория и приложения. — 2022. — Т. 26, № 4. — С. 134–162.
11. Калачев Г. В. Замечания к определению клеточного автомата с локаторами // Интеллектуальные системы. Теория и приложения. — 2020. — Т. 24, № 4. — С. 47–56.
12. Калачев Г. В. Об одновременной минимизации площади, мощности и глубины плоских схем, реализующих частичные булевы операторы // Интеллектуальные системы. Теория и приложения. — 2016. — Т. 20, № 2. — С. 203–266.
13. Калачев Г. В., Титова Е. Е. О мере множества законов движения точки, реализуемых клеточными автоматами // Интеллектуальные системы. Теория и приложения. — 2018. — Т. 22, № 3. — С. 105–125.
14. Кравцов С. С. О реализации функций алгебры логики в одном классе схем из функциональных и коммутационных элементов // Проблемы кибернетики. Вып. 19. — М.: Наука, 1967. — С. 285–293.
15. Кудрявцев В. Б., Алешин С. В., Подколзин А. С. Введение в теорию автоматов. — М.: Изд-во Московского университета, 2019.
16. Кудрявцев В. Б., Гасанов Э. Э., Подколзин А. С. Теория интеллектуальных систем: в 4 кн. Книга четвертая. Теория автоматов. — М.: Издательские решения, 2018.
17. Кудрявцев В. Б., Подколзин А. С., Болотов А. А. Основы теории однородных структур. — М.: Наука, 1990.
18. Кузнецова Е. В. Число состояний универсального автомата бесконечного экрана, реализующего двунаправленное движение на луче // Интеллектуальные системы. Теория и приложения. — 2021. — Т. 25, № 1. — С. 127–147.
19. Кузнецова Е. В. Исследование пограничных случаев реализации клеточным автоматом двунаправленного движения на луче // Интеллектуальные системы. Теория и приложения. — 2021. — Т. 25, № 4. — С. 141–144.
20. Ложкин С. А. О глубине функций алгебры логики в произвольном полном базисе // Вестник МГУ. Серия 1. Математика. Механика. — 1996. — № 2. — С. 80–82.
21. Лупанов О. Б. О синтезе некоторых классов управляющих систем // Проблемы кибернетики. Вып. 10. — М.: Наука, 1963. — С. 63–97.
22. Музаффарова М. Ф. Решение задачи назначения командира клеточными автоматами // Интеллектуальные системы. Теория и приложения. — 2021. — Т. 25, № 4. — С. 181–184.
23. Мур Э. Ф. Математические модели самовоспроизведения // Математические проблемы в биологии. — М.: Мир, 1966.
24. фон Нейман Дж. Теория самовоспроизводящихся автоматов. — М.: Мир, 1971.
25. Сытдыков Т. Р. Сложность синтеза многомерных прямоугольных схем // Интеллектуальные системы. Теория и приложения. — 2019. — Т. 23, № 3. — С. 61–80.
26. Титова Е. Е. Конструирование движущихся изображений клеточными автоматами // Интеллектуальные системы. — 2014. — Т. 18, № 1. — С. 153–180.
27. Arrighi P. An overview of Quantum Cellular Automata // arXiv:1904.12956v2 [quant-ph] 6 Sep 2019
28. Burks A. Essays on Cellular Automata. — University of Illinois Press, 1971.
29. Hochberger C., Hoffmann R. Solving routing problems with cellular automata // Proceedings of the Second Conference on Cellular Automata for Research and Industry. — October 1996. — P. 89–98.
30. Li W. Phenomenology of Non-local Cellular Automata // Stat. Phys. 1 1992. — V. 68, N 5/6. — P. 829–882.
31. Moore F. R., Langdon G. G. A generalized firing squad problem // Information and Control. — March 1968. — V. 12, N. 3. — P. 212–220.
32. von Neumann J. Collected works. — New York, 1961–1963.
33. von Neumann J. Theory of self-reproducing automata. — London, 1966.

Поступило в редакцию 8 IV 2023