ОБОБЩЕННАЯ ВЫЧИСЛИМОСТЬ И ОПРЕДЕЛИМОСТЬ
(Вычислительные системы)
1998 год Выпуск 161
УДК 530.1

# СТАТИСТИЧЕСКИЙ РЕШЕТОЧНЫЙ КОМПЬЮТЕР[1]

Е.И.Латкин[2]

*Резюме.* Рассматривается один вариант архитектуры параллельного компьютера, основанного на "статистических клеточных автоматах", сходных с клеточными автоматами Дж.фон Неймана [1]. Показано, что вычислительные способности статистических клеточных автоматов в точности соответствуют классу функций, алгоритмически вычислимых в обычном смысле.

## Введение

Клеточные автоматы (CA) часто используются для моделирования систем молекул, вовлеченных в химические взаимодействия [2,3]. В работе [4] мы предложили модифицировать машину клеточных автоматов (CAM) таким образом, чтобы она ближе соответствовала именно таким физическим системам.

Интересно, что несмотря на существенное функциональное ограничение, позволяющее статистическим клеточным автоматам учитывать только попарные взаимодействия между "молекулами", изображенными соседними клетками, они все же оказались способны имитировать работу произвольной машины Тьюринга [5,6].

[1] Работа выполнена при поддержке гранта РФФИ № 96-01-00097.

[2] 630090,Новосибирск-90,Пирогова,2,к.202а; http://iele.nsc.ru/~latkin mailto:latkin@iele.nsc.ru

Следовательно, введенная нами "машина статистических клеточных автоматов" (SCAM) способна в принципе служить в качестве модели универсального "химического" компьютера.

В данной работе мы несколько модифицировали определение машины статистических клеточных автоматов, добавив условие "компактности", т.е. потребовали, чтобы в любой момент времени нетривиальными состояниями обладало лишь конечное число ячеек клеточного автомата. Это не препятствует алгоритмической универсальности SCAM и позволяет утверждать также и обратное — что поведение статистических клеточных автоматов алгоритмически вычислимо.

Конечно, алгоритмическая универсальность SCAM показана лишь с точностью до вычислимости "почти наверняка", поскольку "работа" любого статистического клеточного автомата может зациклиться. Однако интуитивно будет ясно из дальнейшего, что вероятность бесконечного цикла для построенного универсального автомата равняется нулю.

## 1. Определения

Рассмотрим бесконечную плоскую решетку, изображающую поверхность кристаллического катализатора, и пронумеруем ее ячейки при помощи пар целых чисел $(i, j) \in \mathbf{Z} \times \mathbf{Z}$. Предположим, что ячейкам, изображающим активные центры на поверхности катализатора, разрешено находиться в любом из $X + 1$ различных "квантовых" состояниях, пронумерованных числами $0, 1, \ldots, X$. Пусть функция $\xi(i, j)$ приписывает каждому "активному центру" $(i, j)$ некоторое, возможно случайное, состояние $\xi(i, j) \in \mathcal{X}$, где $\mathcal{X} = \{0, \ldots, X\}$.

Разумеется, функция $\xi : \mathbf{Z} \times \mathbf{Z} \longrightarrow \mathcal{X}$ должна также зависеть от "физического" времени $t \in \mathbf{R}$.

Эволюция величины $\xi_t(i, j)$ определяется конечным списком "химических процессов" $K_1, \ldots, K_M$. Каждый из процессов в случайно выбираемые моменты времени вовлекает некоторую пару соседних ячеек $(i_0, j_0)$ и $(i_1, j_1)$.

Например, процесс $K_1$ может изображать химическую реакцию между двумя молекулами, адсорбированными на поверхности катализатора и оказавшимися по соседству друг с другом.

Занумеруем процессы при помощи чисел $m = 1, \ldots, M$ и поставим им в соответствие частичные функции $K_m : \mathcal{X} \times \mathcal{X} \longrightarrow \mathcal{X} \times \mathcal{X}$, которые определяют, как изменяются состояния пары соседних ячеек $(i_0, j_0)$ и $(i_1, j_1)$ при соответствующей химической реакции, т.е. формализуют замену текущих состояний $x = \xi(i_0, j_0)$ и $y = \xi(i_1, j_1)$ на модифицированные $x' = \xi'(i_0, j_0)$ и $y' = \xi'(i_1, j_1)$, где $(x', y') = K_m(x, y)$. Функция $K_m(x, y)$ может быть и не определена для некоторых $(x, y)$, если моделируемая реакция не "идет" между молекулами сорта $x$ и $y$.

Понятно, что две различные "реакции" (т.е. функции) $K_m$ и $K_{m'}$ могут диктовать противоречивые инструкции для одной и той же пары соседних ячеек. Более того, даже единственная функция $K_m$ способна противоречиво определять судьбу состояния $\xi(i, j)$, поскольку каждая клетка $(i, j)$ попадает в четыре различные "пары ячеек": совместно с $(i+1, j)$, $(i-1, j)$, $(i, j+1)$ или $(i, j-1)$.

К счастью, попытки противоречивых модификаций ячейки $\xi(i, j)$ не происходят одновременно, также как и физически — две молекулы не могут одновременно адсорбироваться на один и тот же активный центр катализатора. Противоречие разрешает следующий алгоритм метода Монте-Карло [7], который мы используем для вычисления зависимости случайной (!) величины $\xi_t(i, j)$ от времени.

**Алгоритм ($\mathcal{A}_0$).** Алгоритм определяется для моделирования кристалла катализатора конечных размеров, т.е. предполагается, что координаты всех ячеек модельной решетки лежат в диапазоне $0 \le i < I$ и $0 \le j < J$. Также вводятся периодические граничные условия на конечной решетке $(i, j) \in \mathbf{Z}_I \times \mathbf{Z}_J$, т.е. координаты соседней ячейки $(i+1, j)$, $(i-1, j)$, $(i, j+1)$ или $(i, j-1)$ определяются по модулю $I$ или $J$ соответственно. Заметим, что наша об-

резанная поверхность содержит ровно $N = I \cdot J$ ячеек и $2 \cdot N$ пар ячеек.

Определяемый алгоритм метода Монте-Карло аппроксимирует физическое время при помощи дискретной величины $t \in \mathbf{N}$ и коэффициента пропорциональности $\Delta t$. Так называемый *шаг метода Монте-Карло* (или *МК-шаг*) определяет, как меняется состояние модельной решетки за время $\Delta t$. МК-шаг состоит из $2 \cdot N$ элементарных *попыток* локальных трансформаций состояния решетки так, что изменение $\xi_t \longrightarrow \xi_{t+1}$ является результатом цепочки $\xi_t = \xi_t^0 \longrightarrow \ldots \longrightarrow \xi_t^{2N} = \xi_{t+1}$. Давайте определим, как совершается локальная трансформация $\xi_t^n \longrightarrow \xi_t^{n+1}$.

Во-первых, "на поверхности катализатора" $\mathbf{Z}_I \times \mathbf{Z}_J$, где $\mathbf{Z}_L = \{0, \ldots, L-1\}$, случайным образом выбирается пара соседних ячеек $(i_0, j_0)$ и $(i_1, j_1)$, которая будет подвергнута трансформации. Не имеет значения, как эта пара сориентирована в пространстве, т.е. совпадает ли у этих ячеек первая или вторая координаты и прибавлена или отнята единица по другой координате. Затем выбирается случайный номер "элементарного химического процесса" $m \in \{1, \ldots, M\}$, который будет воздействовать на выбранную пару "активных центров"; число $m$ должно быть равномерно распределено от 1 до $M$. Если оказывается, что $K_m(x, y)$ не определено, то это означает, что избранный "химический процесс" не имеет условий при данных состояниях "активных центров" и состояние ячеек остается не модифицированным, т.е. $\xi_t^{n+1} = \xi_t^n$. В противном случае состояния $x = \xi_t^n(i_0, j_0)$ и $y = \xi_t^n(i_1, j_1)$ заменяются на $x' = \xi_t^{n+1}(i_0, j_0)$ и $y' = \xi_t^{n+1}(i_1, j_1)$, где $(x', y') = K_m(x, y)$.

Этот алгоритм моделирует серию случайных термодинамических флуктуаций, активизирующих элементарные физико-химические процессы. В среднем каждая из $2 \cdot N$ пар ячеек затрагивается однажды за МК-шаг. Коэффициент пропорциональности $\Delta t$ как раз соответствует физическому времени, за которое на площадке катализатора из $I \times J$ ячеек в среднем случается $2 \cdot N$ флуктуаций. Впрочем, для наших целей не обязательно от-

слеживать точное соответствие между модельным и физическим временем.

**ОПРЕДЕЛЕНИЕ** 1. Произвольная тройка $(X, \{K_m\}_{m=1}^{M}, \xi_0)$ называется *статистическим клеточным автоматом*, где $X > 0$ — целое число, $\mathcal{X} = \{0, \ldots, X\}$ — конечное множество состояний для ячеек автомата, $\xi_0 : \mathbf{Z} \times \mathbf{Z} \longrightarrow \mathcal{X}$ — начальное состояние всей, очевидно бесконечной решетки в момент времени $t = 0$, и конечный список частичных функций $K_1, \ldots, K_M : \mathcal{X}^2 \longrightarrow \mathcal{X}^2$ есть множество его "элементарных процессов".

**ОПРЕДЕЛЕНИЕ** 2. Пусть $(X, \{K_m\}_{m=1}^{M}, \xi_0)$ — статистический клеточный автомат (в английской аббревиатуре — SCA). Будем говорить, что этот SCA является *компактным*, если $\xi_0(i, j) = 0$ для почти всех ячеек плоскости $(i, j) \in \mathbf{Z} \times \mathbf{Z}$ (т.е. для всех $(i, j)$, кроме некоторого конечного их числа) и для всех $m = 1, \ldots, M$ результат преобразования $K_0(0, 0)$ не определен. (Следовательно, состояние $\xi_t(i, j)$ тривиально для почти всех $(i, j)$ в любой момент времени $t$.)

Поскольку "жизнедеятельность" компактного клеточного автомата разворачивается на уже бесконечной плоскости, нам потребуется расширенный алгоритм, чтобы описать его кинетику. Ниже определяется алгоритм работы компактных статистических клеточных автоматов.

**Алгоритм** $(\mathcal{A}_1)$. Этот алгоритм измеряет модельное время локальными микрошагами $s \in \mathbf{N}$ вместо МК-шагов $t \in \mathbf{N}$, различие между которыми состоит в различной "физической" длительности шага и микрошага. Микрошаг $\xi_s \longrightarrow \xi_{s+1}$ всегда затрагивает только одну пару соседних ячеек, и если бы решетка имела, как и прежде, конечный размер $N = I \times J$, то коэффициенты перерасчета в физическое время $\Delta t$ для шага и $\delta t$ микрошага соотносились бы как $\delta t = \dfrac{\Delta t}{(2 \cdot N)}$ (но $\delta t \neq \text{const}$).

Определим точно, что представляет собой каждая локальная трансформация $\xi_s \longrightarrow \xi_{s+1}$ компактного статистического клеточного автомата. Прежде всего заметим,

что на любом шаге $s \in \mathbf{N}$ состояние $\xi_s(i,j) = 0$ для почти всех $(i,j) \in \mathbf{Z} \times \mathbf{Z}$.

Пусть $L_s$ обозначает число всех таких пар соседних ячеек $x = \xi_s(i_0, j_0)$ и $y = \xi_s(i_1, j_1)$, что $(x,y) \neq (0,0)$. Выберем случайно одну из этих пар, затем выберем случайное число $m \in \{1, \ldots, M\}$. Если $K_m(x,y)$ не определено, то полагаем $\xi_{s+1} = \xi_s$. В противном случае $\xi_{s+1}(i_0, j_0) = x'$ и $\xi_{s+1}(i_1, j_1) = y'$, где $(x', y') = K_m(x,y)$.

Такой микрошаг соответствует смещению физического времени на интервал $\delta t_s = \dfrac{\Delta t}{L_s}$, если коэффициент $\Delta t$ измеряет "временной вес" полного шага метода Монте-Карло по алгоритму $(\mathcal{A}_0)$.

В самом деле, заключим мысленно весь ненулевой фрагмент $\xi_s$ внутрь достаточно большого прямоугольника $-R \leq i,j < R$; тогда процедура преобразования $\xi_s \longrightarrow \xi_{s+1}$ алгоритма $(\mathcal{A}_1)$ в точности соответствует локальному микрошагу $\xi_t^n \longrightarrow \xi_t^{n+1}$ алгоритм $(\mathcal{A}_0)$ для этого прямоугольника. Следовательно, временной вес $\delta t$ этого микрошага равняется доле $\dfrac{\Delta t}{(2 \cdot N)}$, поделенной на вероятность $P = \dfrac{L}{(2 \cdot N)}$ того, что случайно выбранная пара ячеек из прямоугольника $-R \leq i,j < R$ окажется одной из $L$ нетривиальных пар (здесь $N = (2 \cdot R)^2$ — полное число ячеек в этом прямоугольнике). Таким образом, $\delta t = \dfrac{\Delta t}{2N} \cdot \dfrac{2N}{L} = \dfrac{\Delta t}{L}$.

ОПРЕДЕЛЕНИЕ 3. *Машиной статистических клеточных автоматов* (SCAM в английской аббревиатуре) называется класс всех компактных статистических клеточных автоматов, снабженных определенным выше алгоритмом $(\mathcal{A}_1)$ их функционирования.

## 2. Вычислимость SCAM

В этом параграфе мы хотим показать, что поведение любого компактного SCA вычислимо в смысле классической теории рекурсии. Разумеется, поскольку величина

$\xi_s(i,j)$ является случайной, классический компьютер обязан вычислить все варианты эволюции SCA, чтобы описать его поведение.

Вот как это делается.

Закодируем начальное состояние $\xi_0$ компактного клеточного автомата $A = (X, \{K_m\}_{m=1}^{M}, \xi_0)$ и проследим за первым шагом его эволюции $\xi_0 \longrightarrow \xi_1$. Всего существует $L \cdot M$ вариантов поведения автомата $A$ на этом шаге, где $L$ — число нетривиальных пар соседних ячеек в решетке $\xi_0$ и тем самым — число вариантов выбора этой пары, а $M$ — число вариантов выбора из списка "элементарных" процессов $m = 1, \ldots, M$. Обозначим все варианты обновленного состояния решетки после первого микрошага через $\xi_{1;1}, \ldots, \xi_{1;LM}$.

Ясно, что таким образом можно определить вычислимую процедуру построения дерева всех вариантов эволюции автомата $A$ — этаж за этажом; обозначим это дерево через $T_A$. Припишем его корню состояние $\xi_0$, "физическое время" $t = 0$ и "временной вес" $\delta t = \dfrac{\Delta t}{L}$. Первый этаж дерева $T_A$ состоит из узлов $\xi_{1;1}, \ldots, \xi_{1;LM}$, помеченных одним и тем же "физическим временем" $t = \delta t$, но "временной вес" их уже может быть различен, в соответствии с числом нетривиальных пар соседей в состояниях $\xi_{1;1}, \ldots, \xi_{1;LM}$.

Для любого узла $\xi \in T_A$ однозначно определяется его ветвь $b = (\xi_0, \ldots, \xi_s = \xi)$ в дереве $T_A$, и мы можем вычислить "время" $\tau(b) = \delta t_0 + \ldots + \delta t_{s-1}$, соответствующее этому узлу. Мы покажем, что существует только конечное количество всех таких ветвей $b \subseteq T_A$, что соответствующее им время попадает в заданный интервал $t_0 \leq \tau(b) \leq t_1$ для некоторых $t_0, t_1 \in \mathbf{R}$. Значит, все их можно вычислить.

ТЕОРЕМА 1. *Пусть $t_1 \in \mathbf{R}$ — произвольное положительное вещественное число и $A$ — произвольный компактный статистический клеточный автомат. Тогда существует только конечное число таких ветвей $b \subseteq T_A$ в дереве вычислений $A$, что соответствующее "время" $\tau(b) < t_1$.*

ДОКАЗАТЕЛЬСТВО. Обозначим через $\xi_0$ начальное состояние автомата $A$, и пусть $L_0$ обозначает число нетривиальных пар соседей в $\xi_0$. После того, как будет сделан первый шаг алгоритма $(A_1)$, число $L_1$ нетривиальных пар уже в новом состоянии $\xi_1$ не превышает $L_0+3$, так как в $\xi_1$ может появиться не более одной новой нетривиальной ячейки. Таким образом, если построено состояние $\xi_s$, соответствующее ему, то число $L_s$ нетривиальных пар при любом варианте эволюции автомата не превышает $L_0 + 3 \cdot s$.

Вычисляя $\tau(b) = \delta t_0 + \ldots + \delta t_s$, мы обнаружим, что $\delta t_0 \geq \frac{\Delta t}{L_0}$ и соответственно $\delta t_s \geq \frac{\Delta t}{(L_0 + 3s)}$. Таким образом, $\tau(b) \geq f(s)$ для некоторой функции $f(s)$, которая зависит только от "номера этажа" $s = |b|$ ветви $b$ в дереве вычислений $T_A$. Для нас важно, что функцию $f(s)$ можно выбрать монотонной и не ограниченной при $s \longrightarrow \infty$. Поэтому найдется такой номер шага $s \in \mathbf{N}$, что $(\forall s' \geq s)\ f(s) \geq t_1$.

Обозначим через $T_A^s$ поддерево дерева $T_A$, которое содержит все его ветви высотой не более $s \in \mathbf{N}$. По лемме Кенига (König), поскольку дерево $T_A$ — конечно ветвящееся, поддерево $T_A^s$ является конечным. Следовательно, существует не более чем конечное число таких ветвей $b \subseteq T_a$, для которых $\tau(b) < t_1$. Конец доказательства.

Ясно, что мы можем определить следующий способ описания "поведения" компактного SCA в произвольные моменты времени.

ОПРЕДЕЛЕНИЕ 4. Для любого компактного SCA и произвольного неотрицательного $t \in \mathbf{R}$ обозначим через $\xi^t$ конечное распределение всех вариантов состояния $\xi_s$ таких, что время $\tau(\xi_s) \leq t$ и временной вес $\delta t(\xi_s)$ превышает разницу $t - \tau(\xi_s)$.

### 3. SCAM как универсальный компьютер

Мы уже замечали в [5], что SCAM способна имитировать произвольную машину Тьюринга. Вместе с тем мы несколько изменили определение SCAM, добавив огра-

ничение "компактности". Это заставляет заново провести доказательство алгоритмической универсальности SCAM.

Наше доказательство, как и в [5], строго говоря не будет полным. Мы не станем доказывать, что вероятность успешного завершения работы универсального SCA действительно равна единице, поскольку это утверждение представляется интуитивно очевидным.

В работе [9] со ссылкой на [8] указывается, что любая машина Тьюринга может быть промоделирована на некоторой "программируемой машине", которая представляет собой конечный автомат, снабженный двумя регистрами для запоминания натуральных чисел. Для нас будет удобно опереться на конструкцию детерминированного клеточного автомата из [9], который имитирует работу алгоритмически универсальной "программируемой машины". Подобно [5], мы приспособим конструкцию из [9] для статистических клеточных автоматов, на этот раз — компактных.

ТЕОРЕМА 2. *Для любой программируемой машины $\mathcal{M}$ существует такой компактный статистический клеточный автомат $\mathcal{A}$, который имитирует работу программируемой машины $\mathcal{M}$.*

ДОКАЗАТЕЛЬСТВО. Сначала нам придется воспроизвести определение программируемой машины из [9].

ОПРЕДЕЛЕНИЕ 5. *Программируемая машина* $\mathcal{M}$ является тройкой $(\{A,B\},\{C_n\}_{n=1}^{N},i)$, где $\{A,B\}$ — это два регистра, предназначенных для запоминания натуральных чисел, $C_1,\ldots,C_N$ — программа, т.е. последовательность машинных команд, и $i \in \{1,\ldots,N\}$ — номер текущей инструкции.

Список допустимых команд $C_n$, $n=1,\ldots,N$, и правила переходов исчерпываются следующим:

1) $R^+$ — прибавить единицу к регистру $R \in \{A,B\}$;

2) $R^-[m]$ — отнять единицу от регистра $R \in \{A,B\}$, если $m \in \{0,\ldots,N\}$;

3) $[m]$ — перейти к выполнению инструкции номер $m \in \{0,\ldots,N\}$;

4) обычно после того, как выполнена инструкция номер $n \in \{1, \ldots, N\}$, машина переходит к инструкции номер $i = n + 1$. Если при этом оказывается, что $n + 1 > N$, или же требуется выполнить условный или безусловный переход к инструкции номер $m = 0$, такая необходимость рассматривается как команда остановки машины, т.е. по существу является командой HALT.

Вычисления на программируемой машине $\mathcal{M}$ осуществляются в три этапа: 1) в регистры $\{A, B\}$ загружаются начальные данные и полагается $i = 1$; 2) прослеживается эволюция состояния машины до тех пор, пока она не дойдет до инструкции HALT; 3) из регистров $\{A, B\}$ считывается результат вычислений.

Предположим, что нам дана некоторая программируемая машина $\mathcal{M} = (\{A, B\}, \{C_n\}_{n=1}^{N}, i)$. Построим компактный статистический клеточный автомат $\mathcal{A} = (\mathcal{X}, \{K_m\}_{m=1}^{M}, \xi_0)$, имитирующий работу машины $\mathcal{M}$.

Так же, как в [5,9], этот автомат $\mathcal{A}$ будет 1-мерным, т.е. соответствует 1-мерной решетке "квантовых" состояний $\xi_g : \mathbf{Z} \longrightarrow \mathcal{X}$.

Положим число $X$ нетривиальных состояний для ячеек автомата $\mathcal{A}$ равным $N + 15$, где $N$ — число инструкций машины $\mathcal{M}$. Будет удобно обозначить эти состояния мнемоническими символами, тогда их множество $\mathcal{X}$ будет состоять из следующих элементов: 0, HALT, $C_1, \ldots, C_N, a, a^+, a^-, A, A^+, A^-, A^*, b, b^+, b^-, B, B^+, B^-, B^*$.

Начальное состояние решетки $\xi_0 : \mathbf{Z} \longrightarrow \mathcal{X}$ отображает почти все ячейки $i \in \mathbf{Z}$ в $\xi_0(i) =$ "0". В центре решетки располагается "командный центр" автомата, а слева и справа от него мы "нарисуем" регистры $\{A, B\}$ машины $\mathcal{M}$ при помощи символов "A" и "B". В соответствии с этим планом, положим $\xi_0(0) =$ "$C_1$". Далее, положим $\xi_0(-V_0(A)) = \ldots = \xi_0(-1) =$ "$A$", где $V_0(A)$ — есть начальная величина регистра $A$, и $\xi_0(1) = \ldots = \xi_0(V_0(B)) =$ "$B$". Для целей, ясных из дальнейшего, запишем $\xi_0(-V_0(A)-1) =$ "$a$" и $\xi_0(V_0(B)+1) =$ "$b$" на левой и правой "оконечностях" $\xi_0$.

Определим теперь набор "элементарных процессов" $\{K_m\}_{m=1}^{M}$. Будет достаточно одного процесса $M = 1$. Един-

ственная функция трансформаций $K : \mathcal{X}^2 \longrightarrow \mathcal{X}^2$ определяется ниже приведенной таблицей, в которой мы для наглядности пишем $(x, y) \longrightarrow (x', y')$ вместо $K(x, y) = (x', y')$.

Таблица

| **Для всех $C_n = A^+$:** | **Для всех $C_n = B^+$:** |
|---|---|
| $(C_n, a) \to (C_n, a^+)$<br>$(C_n, A) \to (C_n, A^+)$<br>$(C_n, A^*) \to (C_{n+1}, A)$ | $(C_n, b) \to (C_n, b^+)$<br>$(C_n, B) \to (C_n, B^+)$<br>$(C_n, B^*) \to (C_{n+1}, B)$ |
| **Для всех $C_n = A^-[m]$:** | **Для всех $C_n = B^-[m]$:** |
| $(C_n, a) \to (C_m, a)$<br>$(C_n, A) \to (C_n, A^-)$<br>$(C_n, a^-) \to (C_{n+1}, a)$<br>$(C_n, A^*) \to (C_{n+1}, A)$ | $(C_n, b) \to (C_m, b)$<br>$(C_n, B) \to (C_n, B^-)$<br>$(C_n, b^-) \to (C_{n+1}, b)$<br>$(C_n, B^*) \to (C_{n+1}, B)$ |
| **Также (слева):** | **Также (справа):** |
| $(A^+, A) \to (A^+, A^+)$<br>$(A^+, a) \to (A^+, a^+)$<br>$(a^+, 0) \to (A^*, a)$<br>$(A^+, A^*) \to (A^*, A)$ | $(B^+, B) \to (B^+, B^+)$<br>$(B^+, b) \to (B^+, b^+)$<br>$(b^+, 0) \to (B^*, b)$<br>$(B^+, B^*) \to (B^*, B)$ |
| $(A^-, A) \to (A^-, A^-)$<br>$(A^-, a) \to (a^-, 0)$<br>$(A^-, a^-) \to (A^*, a)$<br>$(A^-, A^*) \to (A^*, A)$ | $(B^-, B) \to (B^-, B^-)$<br>$(B^-, b) \to (b^-, 0)$<br>$(B^-, b^-) \to (B^*, b)$<br>$(B^-, B^*) \to (B^*, B)$ |
| **Для всех $C_n = [m]$:** | **Примечания.** |
| $(C_n, a) \to (C_m, a)$<br>$(C_n, b) \to (C_m, b)$<br>$(C_n, A) \to (C_m, A)$<br>$(C_n, B) \to (C_m, B)$ | Если $n = N$, то неявно $C_{n+1}$=HALT.<br>Если $m = 0$, то неявно $C_m$= HALT.<br>Если стрелка в $(x, y) \to (x', y')$ не указана, то $K(x, y)$ не определено. |

Предполагается, что если $K(x, y)$ не определено для некоторого сочетания $(x, y)$, то эта пара просто не обозначается в таблице.

Для примера давайте проследим, как выполняется инструкция $C_n = B^-[m]$ вычитания единицы из регистра $B$, чтобы понять, как работает наш статистический клеточный автомат $\mathcal{A}$.

Предположим сначала, что на некотором шаге $s \in \mathbb{N}$ автомат уже полностью закончил предыдущую инструкцию $C_k$ и текущее состояние регистра $V_s(B) = 0$. В этом случае обязательно $\xi_s(1) = "b"$. Если состояния случайно выбранной согласно алгоритму $\mathcal{A}_1$ пары ячеек $(x, y)$ не равны $(C_n, b)$, то на шаге $s \longrightarrow s+1$ ничего не произойдет. Если же $(x, y) = (C_n, b)$, то автомат $\mathcal{A}$ определяет, что машина $\mathcal{M}$ должна перейти в состояние $\xi_{s+1}(0) = "C_m"$ по команде $(C_n, b) \longrightarrow (C_m, b)$.

Если же текущее состояние $V(B) \neq 0$, то $\xi(1) = "B"$. При первом "успешном" микрошаге, когда пара $(x, y) = (C_n, B)$, срабатывает команда $(C_n, B) \longrightarrow (C_n, B^-)$, и автомат $\mathcal{A}$ "навешивает минус" на первую букву "$B$" справа от центральной "командной" ячейки, полагая в своем обновленном состоянии $\xi(1) = "B^-"$. На следующих успешных шагах маркер "$-$" (минус) будет проставлен последовательно над всеми буквами "$B$", записанными в решетке $\xi : \mathbb{Z} \longrightarrow \mathcal{X}$, — это обеспечивается правилом трансформации $(B^-, B) \longrightarrow (B^-, B^-)$.

На некотором шаге будет помаркирована последняя буква "$B$" и символ "$B^-$" окажется по соседству с концевым символом "$b$". Далее, при помощи правила $(B^-, b) \longrightarrow (b^-, 0)$ концевой символ смещается на одну позицию влево, "съедая" один из символов "$B$", что как раз символизирует вычитание единицы из регистра $B$. На этот момент символ "$b$" и все символы "$B$" (если они еще остались) оказываются помеченными маркером "минус".

Если новое значение $V(B) = 0$, значит, был "съеден" последний символ "$B$" и $\xi(1) = "b^-"$. В этом случае при первом же успешном шаге автомата $\mathcal{A}$ сработает правило

$(C_n, b^-) \longrightarrow (C_{n+1}, b)$ т.е. будет изображен переход машины $\mathcal{M}$ к инструкции номер $n+1$. Заметим, что если при этом $n+1 > N$, то фактически будет смоделирован останов машины $\mathcal{M}$.

Рассмотрим теперь случай, когда новое значение $V(B) \neq 0$. Здесь маркированный символ "$b^-$" соседствует с "$B^-$". При первом успешном шаге сработает правило $(B^-, b) \longrightarrow (B^*, b)$ — маркер "минус" будет снят с концевого символа $b$, и последний из символов "$B$" будет помечен знаком "звездочка", призванным свидетельствовать, что команда $C_n$ уже почти выполнена. На последующих шагах маркер "звездочка" нейтрализует все маркеры "минус" над символами "$B$" в соответствии с правилом $(B^-, B^*) \longrightarrow (B^*, B)$, постепенно подбираясь к центру решетки, пока не достигнет первой ячейки $\xi(1) =$ "$B^*$". На следующем успешном шаге состояние машины $\mathcal{M}$ перевернется в $\xi(0) =$ "$C_{n+1}$" согласно правилу $(C_n, B^*) \longrightarrow$ $\longrightarrow (C_{n+1}, B)$.

Разумеется, кроме "успешных" выборов пары ячеек $(x, y)$ будет встречаться очень много неуспешных, для которых $K(x, y)$ не определено. Таких даже будет большинство, поскольку вероятность успешного выбора на каждом этапе равна всего $\frac{1}{L}$, где $L$ — число нетривиальных пар ячеек для текущего состояния решетки.

Однако неуспешные выборы не портят состояния автомата $\mathcal{A}$. Поэтому вероятность того, что $\mathcal{A}$ не сможет выполнить очередную инструкцию машины $\mathcal{M}$, исчезающе мала: вероятность $k$ неудач подряд равна $\left(1 - \frac{1}{L}\right)^k$, что стремится к 0 при $k \longrightarrow \infty$. Теорема доказана.

## Заключение

Существует по меньшей мере три резона, оправдывающих изучение алгоритмических свойств SCAM.

Во-первых, ее архитектура близка задачам имитационного моделирования реакций гетерогенного катализа, и SCAM может рассматриваться как удобный инстру-

мент для такого моделирования, в котором вычислительные трудности "спрятаны" от прикладного исследователя.

Во-вторых, в силу большей близости SCAM к физической реальности (по сравнению с детерминированными CA, например), можно попытаться, хотя бы умозрительно, соизмерять результаты ее теоретического исследования к реальным химическим процессам.

Наконец, SCAM является реалистичным примером компьютера с "синергетической" архитектурой — надежного вычислительного устройства, состоящего из ненадежных (по крайней мере — не вполне детерминированных в своем поведении) вероятностных элементов.

## Литература


1. NEUMANN John Von. Theory of self-reproducing automata. — Urbana & London: University of Illinois Press, 1966.

2. CHEN S., DAWSON S.P., DOOLEN G.D., JANECKY D.R., LAWNICZAK A. Lattice methods and their applications to reacting systems //Computers Chemical Engineering, 1995. — Vol. 19, № 6/7. — P. 617–646.

3. KAPRAL R. Discrete models for chemically reacting systems //Journal of Mathematical Chemistry, 1991. — Vol. 6. — P. 113–163.

4. ЛАТКИН Е.И. SCAM: гибкая модель гетерогенного катализа //Тезисы 9-й Международной конференции "Математические методы в химии" ММХ-9, Тверь, 1995. — Ч. II. — С. 118–119.

5. ЛАТКИН Е.И. SCAM: химический компьютер //Теория вычислений и языки спецификаций. — Новосибирск, 1995. — Вып. 152: Вычислительные системы. — С. 140–151.

6. ЛАТКИН Е.И. Насколько могут быть сложны каталитические процессы? //Тезисы Второго Сибирского конгресса по прикладной и индустриальной математике ИНПРИМ-96. — Ч. 1. — Новосибирск, 1996. — С. 60.

7. BINDER K. (ed.). Monte Carlo Methods in Statistical Physics. — Berlin: Springer-Verlag, 1979.

8. MINSKY M. Computation: finite and infinite machines. — Englewood Cliffs, NJ: Prentice-Hall, 1967.

9. GOLES E., MAAS A., MARTINEZ S. On the limit set of some universal cellular automata //Theoretical Computer Science, 1993. — Vol. 110. — P. 53–78.

Поступила в редакцию
24 ноября 1997 года