ИСПОЛЬЗОВАНИЕ ИГР КЛЕТОЧНЫХ АВТОМАТОВ ДЛЯ СИНХРОНИЗАЦИИ В РАСПРЕДЕЛЁННЫХ СИСТЕМАХ

М.В. Поникаров,
руководитель проекта, ЗАО «Датавижн СНГ», Нижний Новгород
ponikarov@hse.nnov.ru

Рассмотрена игра клеточных автоматов «FireSquad» с точки зрения применения в качестве формальной модели синхронизации действий в распределённых программных системах. Описаны эвристики, позволяющие сократить пространство вариантов при поиске минимального локального правила.

История развития теории клеточных автоматов

Термин «клеточные автоматы» стал применяться в середине XX в. для обозначения совокупности зависимых элементов с заданными состояниями и правил, в соответствии с которыми состояния этих элементов и зависимости между ними изменяются во времени. Время и состояния при этом дискретны. Использование описанной выше модели для формального моделирования самовоспроизводящихся организмов впервые предложено в работе Фон Неймана [1]. Элементы клеточных автоматов предложено выстроить в одномерные или двухмерные бесконечные прямоугольные таблицы. Состояние элемента изменяется в зависимости от его состояния и от состояния двух (или четырех – для двухмерного случая) ближайших соседей (*см. рис. 1*).

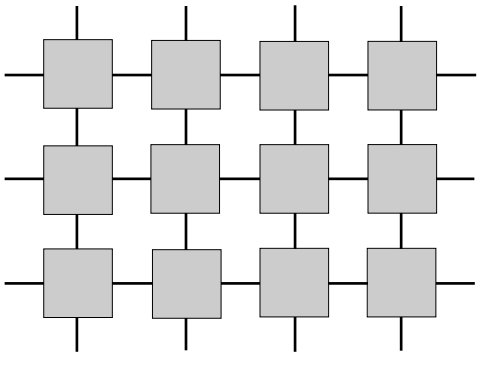

Рис. 1. Схема двухмерного расположения элементов клеточного автомата

Клеточные автоматы в силу своей дискретности сравнительно просто моделируются при помощи ЭВМ и благодаря этому, в 50–70-е гг. XX в. приобретают популярность. Исследователи различных научных областей изучают и используют клеточные автоматы с разнообразными свойствами для различных целей. В это время выходят основные труды, заложившие базис для общей теории клеточных автоматов.

В 1970 г. Bruks [2] в книге «Essays on Cellular Automata» изложил основные термины и положения по этой теме, обобщив множественные теоретические исследования на тот период. В дальнейшем теория разносторонне изучена и расширена Аладьевым [3] (1974 г.), Smith [4] (1976 г.), Vollmar [5] (1977 г.). В этих и других работах изложена «классическая» теория клеточных автоматов – общие положения, теоремы и закономерности, верные для всех направлений исследований на тот период.

Примерно в то же время появились игры для клеточных автоматов – математические модели, имеющие в основе игровое описание. Например, игра «Firing Squad» – задача об одновременном залпе всего орудийного расчёта имеет в основе важную задачу синхронизации [6], а игра «Game of Life» содержит в себе модель поведения популяции в однородной среде [7].

Благодаря своей гибкости и универсальности, клеточные автоматы нашли признание в теории искусственного интеллекта, в самовоспроизводящихся моделях, микро- и макробиологии, вычислениях в среде с возможностью сбоев, в системах

распознавания языка и изображений, в распределённых системах в условиях «информационного голодания».

Начиная с 80-х гг. XX в. изучение клеточных автоматов приняло более специализированный оттенок. На базе общей теории создаются и изучаются различные конфигурации клеточных автоматов для конкретных исследовательских областей. Благодаря разносторонним исследованиям, удалось создать мощную математическую теорию, направленную на классификацию и изучение свойств различных моделей.

В 1983 г. Wolfram издал свою первую работу [8] по статистическим параметрам клеточных автоматов. В дальнейшем Wolfram развил математические аспекты этой теории [9], что позволило ему и другим исследователям подробно классифицировать и изучить практически всё множество клеточных автоматов.

Классификация и свойства клеточных автоматов

Для получения различных конфигураций клеточных автоматов, принято варьировать следующими параметрами:

- ✧ состояния элементов. В каждый момент времени каждый элемент клеточного автомата имеет одно из конечного набора состояний. В зависимости от этих состояний в следующий момент времени набор элементов может принять новое состояние. Если для элементов клеточного автомата множества возможных состояний различны, такой клеточный автомат называется *полигенным*. Но на практике используются ячейки с эквивалентными множествами возможных состояний с алгебраической структурой – линейные клеточные автоматы;
- ✧ геометрия. Элементы могут быть геометрически расположены различным образом. Размерность пространства может быть произвольной, а число элементов – как бесконечным, так и конечным. В последнем случае возникает дополнительная степень свободы в граничных условиях. Они могут быть различными, но на практике используются постоянные во времени (чаще всего – нулевые) или периодические граничные условия. В *динамических* клеточных автоматах геометрия может изменяться со временем, а если геометрия различна на различных участках пространства, такие клеточные автоматы называют *неоднородными*;
- ✧ соседство. Соседи – это элементы, от которых зависит элемент клеточного автомата. Состояние элемента в следующий момент времени вычисляется из состояния самого элемента и его соседей. Соседство в большой степени определяется геометрией клеточного автомата. Для различных целей возможно изменение числа входных состояний элемента. Если для каждого элемента клеточного автомата число входов и выходов одинаковое, такой клеточный автомат называется *сбалансированным*;
- ✧ локальное правило. В соответствии с локальным правилом изменяется состояние элемента клеточного автомата в течение времени. Клеточный автомат, в котором локальные правила различны для различных элементов, называется *разнородным*. Локальное правило может быть недетерминированным, т.е. изменяться во времени или иметь случайную природу.

Варьируя заданные параметры, можно получить клеточные автоматы необходимой конфигурации. Гибкость конфигурации и универсальность вычисления обеспечили высокую популяризацию клеточных автоматов в различных сферах. Произвольность в выборе параметров конфигурации очень удобна для использования, но это придаёт дополнительную сложность в классификации и систематизации знаний теории клеточных автоматов. Тем не менее, наиболее используемо на практике лишь небольшое семейство конфигураций клеточных автоматов. Как правило, каждый из них имеет своё название. Здесь приведён лишь небольшой список наиболее используемых вариантов конфигураций:

- ✧ мозаичный автомат [10]. Клеточный автомат, использующий в локальном правиле каждого элемента не только состояние элемента и его соседей, но и значение общего входного параметра, который может изменяться со временем. Изменение этого параметра ведёт к смене набора правил смены состояний во всём пространстве элементов клеточного автомата. Если из какого-либо начального состояния можно привести клеточный автомат в любую заданную конфигурацию путём варьирования значением общего входного параметра, клеточный автомат называют *полным*;
- ✧ итеративный автомат [11]. Клеточный автомат, в котором лишь один элемент использует для изменения своего состояния значение входного параметра;
- ✧ односторонний клеточный автомат [12]. Такой автомат допускает лишь одностороннее взаимодействие элементов. Например, в одномерном массиве элементов значение каждого элемента зависит лишь от его состояния

и от состояния левого (или правого) соседа. Несмотря на кажущееся вырождение обычного клеточного автомата, односторонние клеточные автоматы достаточно универсальны и используются для распознавания языковых форм;

- ✧ Л-система [13]. Этот тип клеточных автоматов используется для моделирования биологических систем. Это динамические клеточные автоматы (как правило, одномерные или двумерные), в которых с течением времени один элемент может заменяться несколькими или может быть удалённым из системы в соответствии с заданными правилами;
- ✧ отказоустойчивая система [14]. В таких системах моделируется работа клеточных автоматов в реальных условиях: с некоторой вероятностью каждый элемент клеточного автомата может перейти в состояние, не соответствующее локальному правилу. Задачей является создать алгоритмы, для которых работа клеточного автомата будет верной вне зависимости от допущенных ошибок.

Существуют вспомогательные свойства клеточных автоматов, описывающие важные характеристики таких систем. Эти свойства позволяют ввести более детальную классификацию множества клеточных автоматов.

Клеточный автомат называют *инвертируемым*, если существует набор локальных правил, позволяющий однозначно привести клеточный автомат из любого состояния в предыдущее. То есть, если по текущему состоянию всех элементов клеточного автомата можно определить его состояние в предыдущий момент времени.

Состояние всех элементов клеточного автомата (конфигурация клеточного автомата) называется *«Райским садом»*, если такая конфигурация может возникнуть только лишь как начальное значение в любой эволюции клеточного автомата. Отсутствие такой конфигурации означает, что данный клеточный автомат – *эпиморфный*.

Игры для клеточных автоматов

Наряду с практическими моделями, клеточные автоматы могут быть использованы для создания игровых моделей. Игры в большинстве своем представляют лишь более наглядное представление серьёзных практических задач, а изучение игровых проблем позволяет получить более разностороннее и глубокое понимание теории клеточных автоматов в целом. Широко известна игра «Game of Life», моделирующая макроскопическое поведение популяции в однородной среде.

Наше внимание привлекла игра «Fire squad». Это задача, требующая составить локальное правило, одинаковое для каждого элемента одномерного клеточного автомата, удовлетворяющее условиям игры.

В оригинальном описании игры n солдат (один из них – генерал) становятся в одну шеренгу. Каждый солдат может общаться лишь с правым или левым соседом. Генерал даёт команду «старт». Через некоторое время каждый солдат и генерал должны выстрелить одновременно и при этом каждый – в первый раз.

В терминах теории клеточных автоматов, для n элементов, расставленных в ряд и имеющих в качестве соседа лишь правый и левый ближайший элемент, определить общее локальное правило (не зависящее от n), в результате которого клеточный автомат придёт из заданной начальной конфигурации в требуемую. Начальная конфигурация: все элементы имеют «выключенное» состояние (например, «0»), кроме одного – в состоянии «старт» (например, «1»). Требуемая конфигурация: все элементы в состоянии «огонь» (например, – «2»). При этом ни один элемент в промежуточных конфигурациях не должен принимать состояния «2» (*см. рис. 2*).

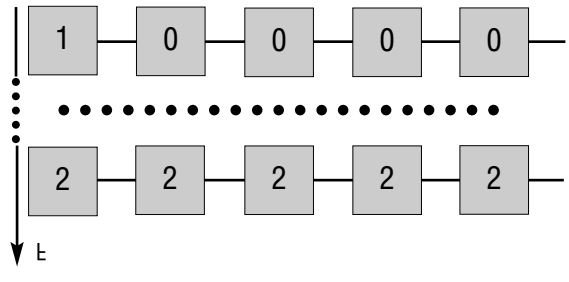

Рис. 2. Игра «Fire squad». Начальное и требуемое состояние

Как видно, данная задача – это задача синхронизации. Она может быть применена в различных прикладных областях. Основная сложность – в неопределённости числа элементов клеточного автомата – солдат.

В общем случае задача многомерная, и генералом может быть любой элемент. Обычно генерал – это один из крайних элементов клеточного автомата. Граничные условия могут быть различны, но принято использовать нулевые граничные условия: крайние элементы используют неизменные нулевые значения состояний несуществующих элементов.

Для двухмерной задачи в начальном состоянии «0» находится $n \cdot m-1$ элементов, образующие n строк и m столбцов, при этом в общем случае $n \neq m$. Генерал может располагаться в произвольном месте. Каждый элемент имеет по 4 соседа. Наиболее популярное решение этой задачи нашёл Minsky [18] – данное решение содержит достаточно много возможных состояний элементов клеточного автомата (99), но обладает важным свойством – такой клеточный автомат инвертированный. Количество шагов, за которое клеточный автомат приходит к состоянию «залпа» равно $3 \cdot n$.

Решение для d-мерной задачи долгое время изучал Szwerinski [19]. Он нашёл решение задачи «Firing squad» с 25 возможными состояниями элементов клеточного автомата.

Практический интерес заключается в минимизации количества временных шагов, предшествующих залпу и (или) минимизации возможных состояний элементов в зависимости от конфигурации клеточных автоматов.

Возможности прикладного применения теории клеточных автоматов

Требование динамической интеграции разнородных компонент в составе единого вычислительного комплекса заставляет вести разработку открытых программных архитектур, включающих большое количество распределённых программных подсистем. Наиболее известными практическими примерами таких архитектур являются многоагентные системы [20, 21] и сети с равноправным правом на хранение информации [22] (peer-to-peer, P2P).

Среди современных теоретических разработок можно указать на направление аморфных вычислений. По мнению авторов [23], в течение ближайших десятилетий две основные нанотехнологии: микропроизводство (microfabrication) и клеточная инженерия – сделают возможным создание систем, включающих мириады очень дешёвых устройств, способных вести обработку различной информации. Причем не обязательно от всех таких устройств требовать гарантии безошибочной работы и определения их точного взаиморасположения в пространстве. Такое изменение технологии приведёт к фундаментальному изменению методов создания и программирования компьютеров, а также само наше представление о вычислениях.

Реализация когерентного поведения большого числа независимых вычислительных сущностей, будь то агенты в многоагентной системе или узлы сети P2P, требует разработки строгих формальных методов для управления транзакциями, принятия коллективных решений и т.д. В независимости от физического размещения отдельных компонентов, их программной реализации разработчик должен быть уверен, что их совместное поведение приведёт к требуемым результатам.

Нами исследована возможность использования формальной модели игры «FireSquad» в качестве такого строгого формального метода координации поведения большого числа независимых программных сущностей.

Подходы к ограничению сложности задачи нахождения требуемого локального правила в игре «FireSquad»

На практике получение локального правила, необходимого для требуемого поведения клеточного автомата нетривиально. Производительность современных вычислительных машин позволяет составлять программы, которые перебором конечного числа вариантов находят локальные правила, удовлетворяющие поставленным критериям.

Рассмотрим основные приемы оптимизации переборной стратегии для игры «Firing squad». Интерес представляет минимизация числа возможных состояний элементов, числа возможных ситуаций для элемента (состояние самого элемента и его соседей) и количества временных интервалов до залпа.

Данная проблема исследуется около 50 лет. В 1957 г. эта задача сформулирована Myhill, а впервые упомянута в изданиях Moore [15]. Очевидно, что минимальный период до залпа равен $T = 2n-1$: «сигнал» должен дойти от генерала до конечного элемента клеточного автомата и обратно, иначе алгоритм не сможет определить число элементов в клеточном автомате. Простые, но в тоже время медленные алгоритмы производят залп за $T = 3n$ шага.

Минимизация возможных состояний – более сложная задача – решается на протяжении всего времени существования этой задачи. В 1966 г. Waksman [16] опубликовал решение одномерной задачи для 16 состояний элементов. В 1967 г. Balzer уменьшил зто число до 8. В 1987 году Mazoyer [17] нашёл решение этой задачи для 6 используемых состояний. Оценку снизу получил Balzer. Он доказал, что решение поставленной задачи невозможно для клеточных автоматов с 4 и менее используемыми состояниями.

Сложность получения минимального набора состояний объясняется большой вычислительной сложностью. Для однозначного определения локального правила для клеточного автомата с k используемыми состояниями необходимо определить k^{k^3} поведения элементов (новое состояние в зависимости

от текущего состояния элемента и его соседей). Для $k = 3$ это число превосходит 10^{12}. При этом каждый из полученных вариантов необходимо проверить на клеточном автомате, что потребует как минимум $T \cdot n$ операций. Такая задача посильна лишь супер ЭВМ.

В ходе работы нами определён ряд эвристик, позволяющих сократить рассматриваемое количество вариантов.

Пусть требуется найти локальное правило, удовлетворяющее задачу «Fired squad» для заданного числа элементов n и числа используемых состояний k. Пусть генерал стоит первым в строю. Локальное правило описывается функцией $F(x, y, z)$, которая возвращает значение нового состояния элемента $(0,1,..., k-1)$ в зависимости от текущих состояний элемента ($x = 0,1,..., k-1$ и состояний его соседей ($y = 0,1,..., k-1$ и $z = 0,1,..., k-1$). Известно, что $F(0, 0, 0) = 0$. Значение функции в других точках необходимо подобрать. В начальном состоянии необходимо определить лишь два значения функции для перехода в следующее состояние:

$$F(1, 0, 0) = a,\ F(0, 1, 0) = b.$$

Для следующего шага – (*см. рис. 3*):

$$F(a, 0, b) = c,\ F(b, a, 0) = d,\ F(b, a, 0) = e.$$

И так далее. Для определения всего множества конфигураций клеточного автомата до залпа необходимо определить лишь требуемую часть значений функции. На каждом шаге необходимо определить лишь 1–4 значений (с течением времени это число уменьшается). Определив, что число шагов до залпа не должно превышать $3n$, получаем снижение числа перебираемых локальных правил порядка k^{8n}.

Также получается выгода: можно найти все возможные локальные правила для заданного n и затем проверять и дополнять полученный набор на $n+1$ и так далее. Начав с небольшого n (например, $n = 5$) можно существенно снизить объём вычислений, а также упростить распределение задачи для параллельного вычисления на нескольких машинах, записывая промежуточный набор локальных правил в файл и считывая его (или часть) на других ЭВМ.

Состояние «2» можно не использовать вовсе. Так как все солдаты должны выстрелить одновременно, можно отслеживать благоприятный момент для залпа: если в определённый момент времени для определения состояния каждого элемента в следующий момент времени необходимо определить новое значение функции, все эти значения можно определить как «2».

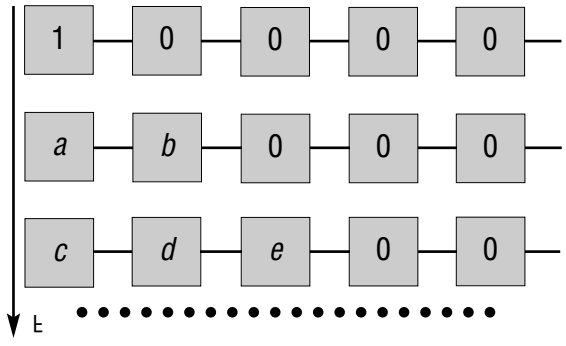

Рис. 3. Изменение конфигурации клеточного автомата с течением времени

Благодаря описанным оптимизациям максимальное число вариантов можно определить как

$$(k-1)^{40} + A,$$

где A – число перебираемых вариантов для других n с уже известным неполным набором правил.

Для $n > 3$ число перебираемых вариантов все ещё остается значительным. Существует ряд оптимизаций, существенно влияющих на скорость нахождения искомого локального правила:

1. Нет смысла определять значение функции $F(x, y, z) > p$, если все предыдущие определённые значения этой функции в других точках не превосходят p более чем на 1 (например, для рисунка 3: $a \le 2$, $b \le a$ и т.д.).

2. Можно обрывать перебор значений функции на шаге $2n$ в случае если к этому моменту времени крайне правый элемент ни разу не изменил своего состояния: в этом случае полученный алгоритм не зависит от числа n: крайне левый элемент не успеет получить информацию от крайне правого элемента.

3. Можно ограничить число определённых значений функции $F(x, y, z)$, если это не противоречит поставленной задаче.

Подавляющее число перебираемых вариантов практически сразу ведёт к затуханию активности клеточного автомата, что приводит к качественному снижению объёма вычислений. Например, при использовании предложенных оптимизаций итоговое число перебираемых вариантов для $n = 4$ получилось порядка 10 миллиардов, что удовлетворяет производительности современного персонального компьютера. Оценка каждого варианта занимает порядка 10000 элементарных операций процессора. Таким образом, время перебора с использованием одного персонального компьютера (процессор Athlon, 1000 мегагерц) для данного варианта занимает около 30 часов.

Таким образом, с появлением высокопроизводительных ЭВМ появилась реальная возможность создавать клеточные автоматы с необходимой конфигурацией, обладающие параметрами, оптимизированными для практической реализации в соответствии с поставленной задачей.

Выводы

В современных распределённых программных системах требуется использование формальных подходов к выработке единого коллективного решения среди большого количества независимых программных сущностей. В данной работе исследованы возможности теории клеточных автоматов и их приложений для разработки таких формальных методов. Модель игры «FireSquad» может служить основой для исследований в этом направлении. Несмотря на большую вычислительную сложность задачи прямого поиска минимальных локальных наборов правил, применение эвристик, полученных в данной работе, позволяет решать задачу для простейших конфигураций за практически приемлемое время.

Однако для использования модели игры «FireSquad» в качестве базового алгоритма координации в распределённых системах требуется решение большого числа проблем. Среди таких проблем укажем лишь на необходимость обобщения игры на случай случайного расположения клеток автомата на плоскости. При этом соседями каждой клетки становятся те, которые находятся в круге некоторого радиуса r. Такая конфигурация типична для аморфных вычислений, однако непосредственное применение известных решений для одномерного и двумерного случая игры «FireSquad» здесь невозможно. ■

Литература

1. Von Neumann, J. and Burks, A. W., Eds, 1966. «Theory of Self-Reproducing automata». University of Illinois Press, Champaign, IL.
2. Burks, A., Ed. 1970. «Essays on Celluar Automata». University of Illinois Press, Champaign, IL.
3. Aladyev, V. 1974. «Survey of research in the theory of homogeneous structures and their applictions». Math. Biosci. 15, 121–154.
4. Smith III, A. 1976. «Introduction to and survey of polyautomata theory». In «Automata, Languages, Development». North-Holland Publishing Co., Amsterdam, The Netherlands.
5. Vollmar, T. 1977. «Cellular spaces and parallel algorithms, and introductory survey». In «Parallel Computation-Parallel Mathematics», M. Feilmeier, Ed. North-Holland Publishing Co., Amsterdam, The Netherlands. 49–58.
6. Moore, E., Ed. 1964. «Sequential Machines. Selected Papers». Addison-Wesley Publishing Co., Inc., Redwood City, CA.
7. Gardner, M. 1970. «The Fantastic combinations of John Convay's new solitaire game of «Life». Sci. Am. 223, 120–123.
8. Wolfram, S. 1983. «Statistical mechanics of cellular automata». Rev. Modern Phys. 55, 601–644.
9. Wolfram, S. 1986. «Theory and Applications of Cellular Automata: Including Selected Papers 1983–1986». World Scientific Publishing Co., Inc., River Edge, NJ.
10. Yamada, H. and Amoroso, S. 1969. «Tesselation Automata». Inf. Control 14, 299–317.
11. Seiferas, J. 1982. «Observations of nondeterministics multidimensional iterative arrays» In «Proceedings of the ACM Symposium of the Theory of Computing», ACM Press, New York, NY, 276–289.
12. Chang, J. H., Ibarra, O. H., and Vergis, A. 1988. «On the power of one-way communication». J. ACM 35, 3 (July 1988), 697–726.
13. Lindenmayer, A. 1968. «Mathematical models for cellular automata and algebraic series».Theor. Comput. Sci. 119, 2 (Oct. 25, 1993), 345–354.
14. Nishio, H. and Kobuchi, Y. 1975. «Fault tolerant cellular spaces». J. Comput. Syst. Sci. 11, 150–170.
15. Moore, E. and Langdon, G. 1968. «A generalized firing squad problem». Inf. Control 12, 212–220.
16. Waksman, A. 1966. «An optimum solution to the firing squad synchronization problem». Inf. Control 9, 67–78.
17. Mazoyer, J. 1987. «A six-state minimal time solution to the firing squad synchronization problem». Theor. Comput. Sci. 50, 2 (Sept. 1987), 183–238.
18. Minsky, M. 1972. «Computation: Finite and Infinite Mashines». Prentice Hall.
19. Szwerinski, H. 1982. «Time-optimal solution of the firing-squad-synchronization-problem for n-dimensional rectangles with the general at arbitrary position». Theoretical Computer Science, 19, 305–320.
20. Cockayne W.T., Zyda M. Mobile Agents. – Manning Publ.Co, 1998. –312 pp.
21. Трахтенгерц Э.А. Компьютерная поддержка принятия решений. – М: Синтег, 1998. – 376 с.
22. Parameswan M., Susarla A., Whinston A.B. P2P Networking: An Information-Sharing Alternative // Computer, №7, 2001. – pp. 31–38.
23. Abelson H., Allen D., Coore D. et all., Amorphous Computing // Communications of the ACM , May 2000-volume 43, №5. – pp.74–82.