На правах рукописи

Наумов Лев Александрович

Метод введения обобщенных координат и инструментальное средство для автоматизации проектирования программного обеспечения вычислительных экспериментов с использованием клеточных автоматов

Специальность 05.13.12. Системы автоматизации проектирования (приборостроение)

АВТОРЕФЕРАТ
диссертации на соискание ученой степени
кандидата технических наук

Санкт-Петербург – 2007

Работа выполнена в Санкт-Петербургском государственном университете информационных технологий, механики и оптики

Научный руководитель	доктор технический наук, профессор Шалыто Анатолий Абрамович
Официальные оппоненты	доктор технических наук Бухановский Александр Валерьевич
	кандидат физико-математических наук, старший научный сотрудник Новиков Фёдор Александрович
Ведущая организация	Санкт-Петербургский государственный электротехнический университет "ЛЭТИ" имени В.И. Ульянова (Ленина)

Защита диссертации состоится 22 мая 2007 года в 15 часов 50 минут на заседании диссертационного совета Д 212.227.05 при Санкт-Петербургском государственном университете информационных технологий, механики и оптики по адресу: 197101, Санкт-Петербург, Кронверкский пр., д. 49.

С диссертацией можно ознакомиться в библиотеке СПбГУ ИТМО.

Автореферат разослан 11 апреля 2007 г.

Ученый секретарь совета Д 212.227.05,
кандидат технических наук, доцент — Поляков Владимир Иванович

Общая характеристика работы

Актуальность проблемы. В настоящее время большинство вычислительных задач требует для своего решения ресурсов, превосходящих возможности однопроцессорных компьютеров. Появляются алгоритмы, использующие параллельные или распределенные вычисления. Кроме того, возникают новые задачи, которые требуют параллельной или распределенной архитектуры для своего решения.

Принимая во внимание то обстоятельство, что в настоящее время при разработке сложных программно-аппаратных комплексов значительная доля затрат идёт на создание программного обеспечения, а также то, что это доля неуклонно увеличивается, необходимость создания инструментального средства для автоматизации проектирования программного обеспечения систем с параллельной и распределенной вычислительными архитектурами становится актуальной.

Существует ряд математических моделей параллельных и распределенных вычислений. Одной из них являются клеточные автоматы, применению которых и посвящена настоящая диссертация.

Клеточные автоматы – это дискретные динамические системы, поведение которых может быть полностью описано в терминах локальных зависимостей. Эти автоматы представляют собой модели параллельных вычислений, которые обладают сколь угодно большой степенью параллелизма. Можно считать, что они представляют собой дискретный эквивалент понятия "поле". Клеточные автоматы применяются для проведения различных вычислительных экспериментов, так как они удобны, например, для численного решения дифференциальных уравнений и уравнений в частных производных. Они также широко используются для моделирования поведения сложных систем.

Существует ряд средств автоматизации проектирования программного обеспечения вычислительных экспериментов с использованием клеточных автоматов таких, как, например, средство *SIMP/STEP* (реализующее набор функций для решения задач физического моделирования), *CAGE* (инструмент для образовательных целей) или *Mirek's Cellebration* (*MCell*) (средство для визуализации простых алгоритмов) и *CDL* (инструмент, позволяющий использовать *FPGA*-системы для выполнения вычислительных экспериментов). Однако все эти средства имеют ограничения, которые накладываются на предметные области решаемых задач, реализуемые клеточные автоматы или используемые вычислительные архитектуры. Не существует единого средства, пригодного, как для образовательных, так и для научных целей. Поэтому для автоматизации проектирования программного обеспечения вычислительных экспериментов на основе клеточных автоматов необходимо разработать универсальное, расширяемое инструментальное средство, обладающее широким спектром функциональных возможностей, так как существующие инструменты не позволяют решать задачи из различных предметных областей с помощью разнообразных клеточных автоматов. Так, например, далеко не каждое средство позволяет моделировать автоматы, как с двумерными, так и с трехмерными решетками. Малая их часть поддерживает использование разнообразных окрестностей. Например, окрестность Марголуса поддерживают всего несколько специализированных средств. Не одно из них не допускает хранения строк в клетках, что необходимо при реализации задач параллельного синтаксического анализа. Только некоторые позволяют описывать структуры данных, хранимые в клетках.

Отсутствие универсальных средств во многом связано с тем, что до настоящего времени не был предложен метод введения обобщенных координат, который позволял бы обеспечить единообразие представления данных о состояниях клеток автоматов с различными решетками (в том числе, и с решетками разной размерности).

Настоящая диссертация посвящена созданию такого метода, а также универсального инструментального средства на его базе. Такое средство позволит использовать естественный параллелизм клеточных автоматов и может быть применено для реализации, визуализации и изучения параллельных и распределенных алгоритмов, которые, например, используются при создании и применении систем автоматизации проектирования приборов. Оно заменит

дорогостоящие экспериментальные установки для проведения экспериментов. Кроме того, необходимо отметить, что сложность современного параллельного программного обеспечения такова, что средство автоматизации его проектирования необходимо.

Из изложенного следует, что работа по созданию метода введения обобщенных координат, обеспечивающего единообразие представления данных о состояниях клеток разнообразных решеток клеточных автоматов, а также удобного, универсального, расширяемого средства автоматизации проектирования программного обеспечения вычислительных экспериментов с использованием клеточных автоматов на его основе является весьма актуальной.

Целью диссертационной работы является разработка метода введения обобщенных координат и создание на его основе инструментального средства автоматизации проектирования программного обеспечения вычислительных экспериментов с использованием клеточных автоматов, которое позволит проводить эксперименты на различных вычислительных архитектурах.

Основные задачи диссертационной работы состоят в следующем:

1. Разработка метода введения обобщенных координат для решеток клеточных автоматов.
2. Разработка на основе метода введения обобщенных координат инструментального средства, включающего в себя среду выполнения и библиотеку для создания клеточных автоматов, которое предназначено для автоматизации проектирования программного обеспечения вычислительных экспериментов с их использованием.
3. Применение инструментального средства для автоматизации проектирования программного обеспечения ряда вычислительных экспериментов с использованием клеточных автоматов на различных вычислительных архитектурах.
4. Классификация всех структур, которые порождаются одномерными клеточными автоматами из точечного зародыша с целью анализа простейших клеточных автоматов, демонстрирующих сложное поведение (в том числе, самовоспроизведение).

Научная новизна состоит в том, что впервые предложен метод введения обобщенных координат, обеспечивающий единый подход для хранения данных из разнообразных решеток клеточных автоматов, на базе которого создано инструментальное средство для автоматизации проектирования программного обеспечения вычислительных экспериментов с использованием клеточных автоматов.

Положения, **выносимые на защиту**:

1. Метод введения обобщенных координат для клеточных автоматов обеспечивает единообразие представления данных для автоматов с различными решетками, в том числе, и разной размерности.
2. Классификация всех структур, порождаемых одномерными клеточными автоматами из точечного зародыша, группирует все типы поведения, порождаемые простейшими клеточными автоматами (в том числе, самовоспроизведение) в классы эквивалентности.

Методы исследования. В работе использованы методы теории автоматов, теории множеств, теории параллельных вычислений, теории алгоритмов, вычислительной математики, основы объектно-ориентированного программирования.

Достоверность и обоснованность научных положений, выводов и практических рекомендаций, полученных в диссертации, подтверждается доказательствами, обоснованием постановок задач, точной формулировкой критериев, компьютерным моделированием, а также результатами использования методов, предложенных в диссертации, в ходе выполнения вычислительных экспериментов.

Практическое значение работы состоит в том, что разработано универсальное и расширяемое инструментальное средство для автоматизации проектирования программного обеспечения вычислительных экспериментов с использованием широкого спектра клеточных автоматов, которое позволяет проводить вычислительные эксперименты на однопроцессорных, многопроцессорных и кластерных платформах.

Реализация результатов работы (имеются акты внедрения). Для образовательных целей результаты используются в Санкт-Петербургском государственном университете

информационных технологий, механики и оптики (СПбГУ ИТМО), в Санкт-Петербургском государственном университете и в Университете Амстердама (Нидерланды). Для научно-исследовательских целей результаты используются в СПбГУ ИТМО, в Университете Амстердама и в Политехническом университете Бухареста (Румыния).

Научно-исследовательские работы. Результаты диссертации получены в ходе выполнения в СПбГУ ИТМО научно-исследовательских работ по теме "Разработка технологии автоматного программирования", выполненной по гранту РФФИ № 02-07-90114 в 2002-2003 годах; по теме "Разработка технологии программного обеспечения систем управления на основе автоматного подхода", выполняемой по заказу Министерства образования РФ в 2002-2005 годах; по гранту конкурса персональных грантов для студентов и аспирантов вузов Санкт-Петербурга в 2004 году; по теме "Разработка среды и библиотеки "*CAME&L*" для организации параллельных и распределенных вычислений на основе клеточных автоматов", выполненной по гранту РФФИ № 05-07-90086 в 2005-2006 годах. При выполнении работ по этому гранту автор был ответственным исполнителем. Научно-исследовательская деятельность автора дважды отмечена стипендией Президента РФ.

Апробация результатов работы. Основные положения диссертационной работы докладывались на международной научной конференции "*International Conference of Computational Sciences*" (Санкт-Петербург – Мельбурн, 2003 г.), на международной научной конференции "*Cellular Automata for Research and Industry*" (Амстердам, 2004 г.), на научной школе "*Joint Advanced Students School*" (Санкт-Петербург, 2004 г.), на научно-методических конференциях "Телематика-2004", "Телематика-2005", "Телематика-2006" (Санкт-Петербург), на научной школе "*Ferienakademie*" (Южный Тироль, 2005 г.), на всероссийской научной конференции "Научный сервис в сети Интернет: технологии распределенных вычислений" (Дюрсо, 2005 г.), на международной научной конференции "Высокопроизводительные параллельные вычисления на кластерных системах" (Санкт-Петербург, 2006 г.).

Публикации. По теме диссертации опубликовано 14 печатных работ, в том числе, в журналах "Известия РАН. Теория и системы управления" (входит в перечень ВАК), "Информационно-управляющие системы", "Телекоммуникации и информатизация образования", а также в материалах конференций "*International Conference of Computational Sciences*", "*Cellular Automata for Research and Industry*", "Телематика" и "Научный сервис в сети Интернет: технологии распределенных вычислений".

Структура диссертации. Диссертация изложена на 283 страницах и состоит из оглавления, введения, четырех глав, заключения, списка терминов, предметного указателя, списка литературы и двух приложений. Список литературы состоит из 97 наименований. Работа содержит 44 рисунка и 13 таблиц.

Содержание работы

Во введении описывается объект и предмет исследования, ставятся цели и задачи исследования, обосновывается актуальность темы диссертационной работы. Дается оценка новизны полученных результатов, формулируются положения, выносимые на защиту.

Первая глава содержит введение в теорию клеточных автоматов и исторический обзор их практического использования в различных предметных областях. Кроме того, в главе предложена компонентная модель декомпозиции клеточных автоматов.

Диссертация развивает исследования в области клеточных автоматов таких учёных, как, например, Джон фон Нейман, Томас Уорш, Йорг Веймар и Томазо Тоффоли.

На основе изучения литературы и опыта проведения экспериментов производится анализ инструментальных средств автоматизации проектирования программного обеспечения вычислительных экспериментов с использованием клеточных автоматов и требований к таким средствам. Установлено, что инструментальное средство должно:

1. Предоставлять пользователям дружественный, удобный, современный интерфейс с богатым набором возможностей для работы.
2. Наглядно визуализировать состояния решетки, обеспечивать удобную навигацию по ней.
3. Быть универсальным и не накладывать ограничений множество реализуемых автоматов.

4. Выполнять итерации, используя современные вычислительные и коммуникационные технологии.
5. Оптимизировать вычисления, например, исключать не обновляемые зоны решетки, использовать шаблоны для сравнения окрестностей и т.п.
6. Удобно и надежно проводить продолжительные вычислительные эксперименты, продолжать постоянно быть интерактивным и отвечать на запросы пользователя.
7. Позволять производить не только вычисления, но и анализ их результатов, поддерживать построение графиков, демонстрировать изменения параметров системы и т.п.

В работе рассматриваются более двадцати существующих инструментальных средств, приводятся их краткие характеристики, ссылки на сайты или статьи, в которых они были описаны, условия распространения и системные требования. Результаты их сравнения приведены в таблице 1.

Таблица 1. Сравнительные характеристики рассматриваемых инструментальных средств

Название	*CAGE*	*CAM Simulator*	*CAMEL*[1]	*CANL*	*CAPow*	*CASim*	*CAT/ CARP*	*CDL*	*CDM/ SLANG*	*CeLLab*	*CELLAS*	*Cellsim*	*Cellular/ Cellang*	*CEPROL*	*DDLab*	*HICAL*	*LCAU*	*MCell*	*SCARLET*	*SIMP/ STEP*	*WinCA*
1	+	–	–	+	+	–	–	–	–	–	–	+	+	–	+	+	–	+	+	–	+
2	+	+	–	–	+	–	–	–	–	–	–	+	–	–	+	–	+	+	+	–	+
3	–	–	+	–	–	–	–	–	–	–	–	–	+	–	–	–	–	–	–	–	–
4	–	–	+	+	–	–	–	+	–	+	–	–	–	–	–	–	–	–	–	+	–
5	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	+	–
6	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	–
7	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	+	+	–	–	–	–

"+" – свойство выполняется в должной мере <под таблицей>; "–" – свойство в должной мере не выполняется.

Таким образом, ни одно из перечисленных инструментальных средств не только не удовлетворяет всем требованиям, но даже не поддерживает больше трех из них. Малое число средств обладает универсальностью, поддерживает вычисления на параллельных и распределенных архитектурах, оптимизацию вычислений и анализ результатов. В то время, как для проведения современных научных экспериментов наглядная визуализация, универсальность, поддержка вычислений на параллельных и распределенных архитектурах, надежность и анализ результатов являются необходимыми.

Для обеспечения выполнения всех указанных требований предлагается использовать модель декомпозиции клеточных автоматов, в соответствии с которой клеточный автомат должен быть реализован, как совокупность четырех взаимодействующих компонентов:

- решетки (*grid*), осуществляющей визуализацию автомата и навигацию по его клеткам;
- метрики (*metrics*), которая сопоставляет клеткам их координаты и вводит отношение соседства и функции вычисления расстояний между клетками по разным направлениям;
- хранилища данных (*datum*), обеспечивающего хранение, обмен и преобразования состояний клеток, а также некоторые аспекты визуализации (соответствие между состоянием клетки и цветами, используемыми для ее отображения);
- правил (*rules*), описывающих итерацию (не только функцию переходов, но и метод разделения задачи на подзадачи, оптимизацию вычислений и т.д.).

Помимо компонентов этих четырех типов в эксперименте должна поддерживаться возможность использования вспомогательных компонентов – анализаторов (*analyzer*),

[1] Это средство называется почти так же, как и предложенный в настоящей диссертации инструмент *CAME&L* (*Cellular Automata Modeling Environment & Library*). Амперсант в названии последнего появился именно в связи с существованием проекта "*CAMEL*" ("*Cellular Automata environMent for systEms modeLing*"). Кроме похожих названий инструментальные средства не связывает больше ничего.

позволяющих наблюдать за изменениями (строить графики, составлять файлы отчетов и т.п.) ключевых параметров эксперимента.

Каждый компонент декларирует список собственных параметров, изменение значений которых обеспечивает настройку компонента.

Такая модель компонентной декомпозиции клеточных автоматов позволяет "собирать" автоматы с различной функциональностью, а за счет небольших изменений переходить от одной задачи к другой, решать одну и ту же задачу на различных решетках и различных вычислительных архитектурах, использовать различные метрики и т.д. Кроме того, она предоставляет возможность распределять разработку программного обеспечения эксперимента между разными исполнителями.

Одним из основных недостатков известных подходов к реализации клеточных автоматов, является то, что не был известен метод, обеспечивающий единообразие представления данных для различных клеточных автоматов.

Таким образом, анализ существующих средств автоматизации проектирования программного обеспечения вычислительных экспериментов с использованием клеточных автоматов, проведенный в первой главе, выявил отсутствие метода, обеспечивающего единообразие представления данных для различных решёток клеточных автоматов, и инструмента, удовлетворяющего всем требованиям, перечисленным выше.

Построена компонентная модель декомпозиции автоматов, для дальнейшего создания инструментального средства и проведения экспериментов на его основе.

Во второй главе предлагается метод введения обобщенных координат для решеток клеточных автоматов, который основан на нумерации всех клеток решетки неотрицательными целыми числами. Обобщенной координатой названо неотрицательное целое число, однозначно указывающее на клетку решетки автомата произвольной размерности. Необходимо отметить, что понятие "обобщенные координаты" было введено автором диссертации.

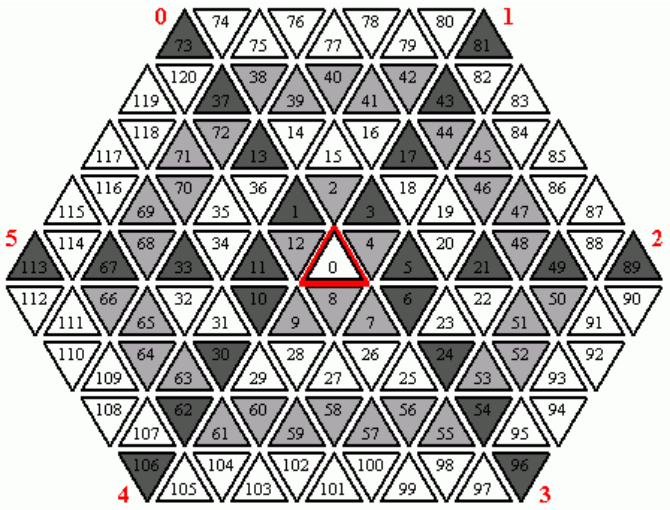
Одной из клеток присваивается номер ноль, она принимается за начало координат. Номера остальных клеток характеризуют их положение относительно него.

В общем случае введение обобщенных координат заключается в построении взаимнооднозначного отображения из множества неотрицательных целых чисел во множество целых чисел в степени k. В настоящей главе рассматриваются различные отображения для k=2, однако предлагаемые методы могут быть применены и для решеток большей размерности.

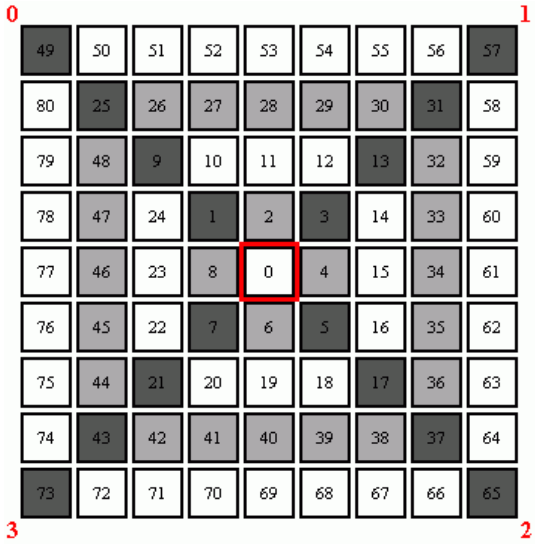
Для того чтобы при выполнении итераций не пришлось преобразовывать обобщенные координаты в декартовы и обратно, необходимо разработать математический аппарат, позволяющий для каждой клетки находить ее соседей, не переходя в другую систему координат. В силу локальности взаимодействий, для нахождения соседей произвольной окрестности достаточно обеспечить нахождение лишь непосредственных соседей. В случае сложной окрестности, ее элементы можно будет найти при помощи рекурсии.

Метод иллюстрируется примерами для двумерных автоматов, для которых вводятся:

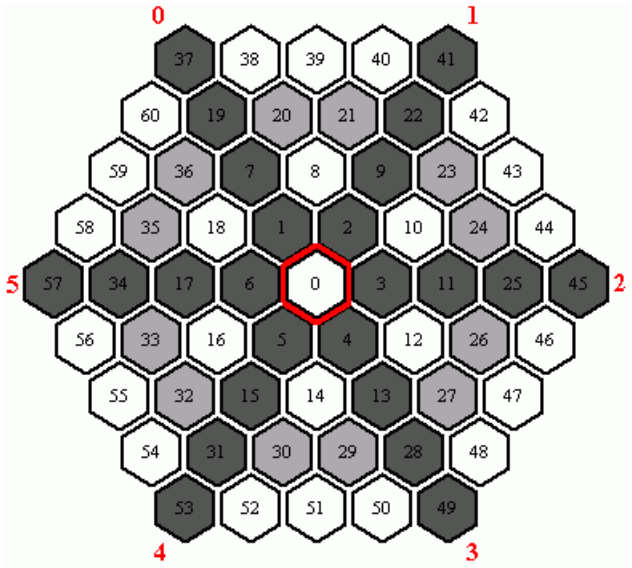
- спиральные обобщенные координаты для решеток из треугольников (рис. 1.а);
- спиральные обобщенные координаты для решеток из квадратов (рис. 1.б);
- спиральные обобщенные координаты для решеток из шестиугольников (рис. 1.в);
- координаты для решеток из треугольников, базирующиеся на спиральных обобщенных координатах для решеток из шестиугольников (рис. 1.г);
- обобщенные координаты для решеток из квадратов, основанные на кривых Пеано (рис. 1.д).



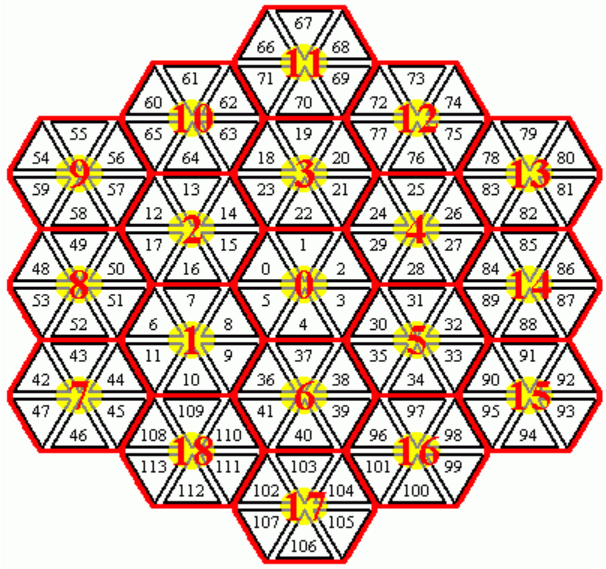
а. Спиральные обобщенные координаты для треугольной решетки



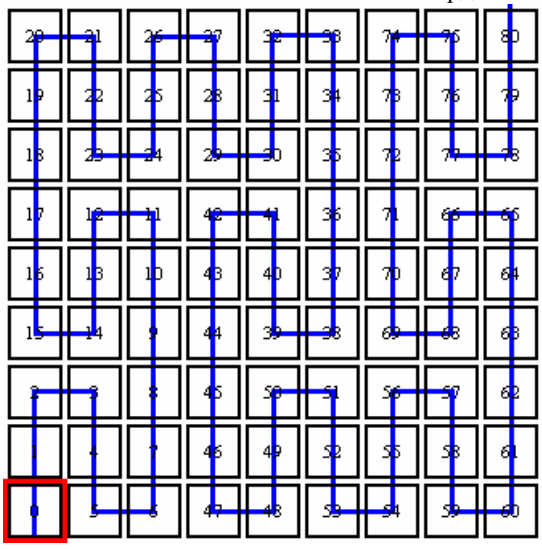
б. Спиральные обобщенные координаты для квадратной решетки



в. Спиральные обобщенные координаты для шестиугольной решетки



г. Обобщенные координаты для треугольной решетки, базирующиеся на спиральных обобщенных координатах для шестиугольной решетки



д. Обобщенные координаты для квадратной решетки, основанные на кривой Пеано

Рис. 1. Примеры введения обобщенных координат

Приняв за единицу измерения производительности средние показатели самого медленного метода – спиральных координат для треугольной решетки, экспериментально установлено, что реализация спиральных координат для квадратной решетки функционирует в 3,9 раза быстрее, реализация обобщенных координат для треугольной решетки, базирующихся на спиральных координатах для шестиугольной – в 4,4 раза быстрее, а реализация спиральных координат для шестиугольной решетки – быстрее в 6,2 раза.

Перечислим основные преимущества использования обобщенных координат.

1. Обобщенные координаты обеспечивают универсальность организации данных, позволяя хранить состояния различных решеток клеточных автоматов в одномерном массиве данных. При переходе от автомата с одной решеткой к автомату с другой необходимо изменить только набор функций для определения непосредственных соседей клетки. В случае реализации рассматриваемого подхода с помощью инструментального средства, созданного в соответствии с компонентной моделью декомпозиции клеточных автоматов, предложенной в первой главе, смену набора функций легко обеспечить посредством выбора другого компонента метрики.
2. Существенно более удобным становится изменение размеров решетки. Расширение одномерного массива за счёт записи данных в его конец значительно проще, чем изменение размера и перераспределение многомерной структуры.
3. Одномерный массив можно рассматривать, как последовательность данных, а последовательность данных удобно сериализовывать и хранить.
4. Обобщенные координаты – это идея, которая может быть адаптирована под конкретную задачу с учетом, например, внутренней симметрии моделируемой системы и т.п.
5. Отсутствие существенной зависимости от положения нуля позволяет перемещать его в соответствии с условиями задачи. Например, при моделировании может возникнуть необходимость поместить ноль в центр активности для того, чтобы оперировать там с меньшими числами. С другой стороны, можно отодвинуть его дальше от центра активности для того, чтобы основные вычисления происходили на длинных ребрах колец, где получение координат некоторых соседей выполняется за счет прибавления или вычитания единицы из координаты клетки.
6. Для некоторых решеток понятие декартовых координат неоднозначно. Например, для решетки шестиугольников неясно, почему клеткам необходимо присваивать одинаковые координаты вдоль той или иной оси, когда их центры не лежат на одной прямой параллельной этой оси. Для более "вычурных" решеток подобные проблемы еще актуальнее.

Обсуждаемые примеры введения обобщенных координат были апробированы при проведении вычислительных экспериментов.

Таким образом, во второй главе предложен метод введения обобщенных координат, который позволяет обеспечить единообразие организации данных о состояниях клеток разнообразных решеток клеточных автоматов, а также обладающий рядом других важных свойств.

Третья глава описывает реализацию обобщенных координат в предложенном инструментальном средстве *CAME&L* (*Cellular Automata Modeling Environment & Library*). Рассматриваются три его важнейшие составные части:

- Среда выполнения – приложение с ясным и дружественным пользовательским интерфейсом. Базовой абстракцией в ней является понятие "эксперимент" – вычислительная задача, описанная в терминах клеточных автоматов. Среда может работать на однопроцессорном компьютере, многопроцессорной системе или в вычислительном кластере. Она имеет многодокументный интерфейс (*Multi-Document Interface*) для того, чтобы управлять несколькими экспериментами одновременно, а также реализовывать межавтоматные взаимодействия в процессе моделирования. Помимо этого она обеспечивает сохранение документов в формате *XML* (файлы с расширением "*camel*"), что позволяет пользователям хранить и обмениваться описаниями экспериментов. Для

уменьшения размера этих файлов информация о состоянии решетки автомата, может быть сжата внутри документа с помощью алгоритма *BZip2*. Рассматриваются основные принципы работы в среде, назначения всех пунктов меню, формат файлов и т. д.

- Стандартные компоненты – базовый набор решеток, хранилищ данных, метрик, правил и анализаторов, из которых могут быть построены решения пользовательских задач.
- Библиотека для разработки клеточных автоматов *CADLib* ("*Cellular Automata Development Library*") – набор *C++*-классов, которые предназначены для создания пользователями новых компонентов, предназначенных для решения их задач. Приводятся описания большинства классов, функций, макроопределений и констант, содержащихся в библиотеке. Излагаются особенности использования библиотеки для создания компонентов.

Структурная схема, показывающая основные составные части предложенного инструментального средства и эксперимента, выполняемого в нем, приведена на рис 2.

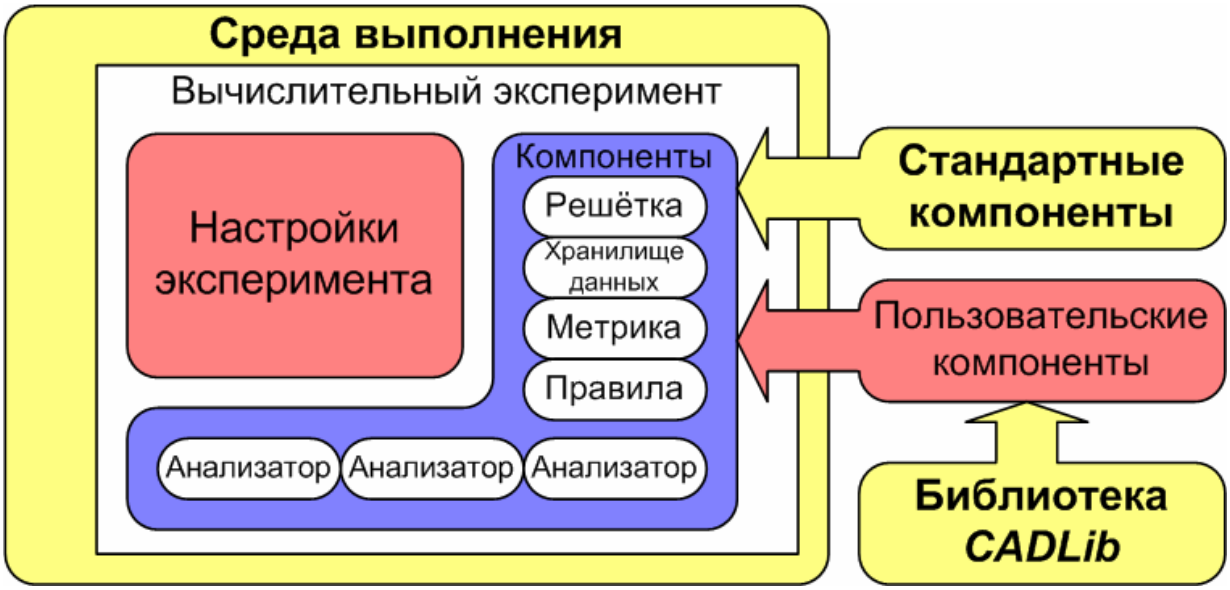


Рис. 2. Структурная схема предложенного инструментального средства и эксперимента в нем. Жирным шрифтом на рисунке показаны три основные части инструментального средства *CAME&L*. Вычислительный эксперимент, выполняемый в среде, характеризуется набором выбранных компонентов и установленными настройками. В набор компонентов может входить произвольное количество анализаторов, а также решетка, хранилище данных, метрика и правила в единственном экземпляре. Компоненты выбираются из числа стандартных (поставляемых в составе инструментального средства) или пользовательских (разработанных с помощью библиотеки *CADLib*).

В главе описывается также использование компонентной модели декомпозиции клеточных автоматов при разработке пользовательских компонентов для проведения вычислительных экспериментов. Все стандартные компоненты создавались аналогично.

Таким образом, было создано универсальное, расширяемое инструментальное средство, удовлетворяющее всем семи требованиям, перечисленным в первой главе, которое может эффективно использоваться, как для научных экспериментов, так и для образовательных целей. Средство позволяет организовывать эксперименты на однопроцессорной, многопроцессорной и кластерной вычислительных архитектурах. При этом для взаимодействия вычислительных узлов кластерных систем автором предложен специализированный сетевой протокол *Commands Transfer Protocol*, который обеспечивает быстрый обмен короткими сообщениями и имеет, при этом, простую реализацию.

В четвертой главе приводятся результаты использования рассматриваемого инструментального средства для автоматизации проектирования программного обеспечения вычислительных экспериментов.

Первый из них был проведён для тестирования предложенного средства на однопроцессорной, многопроцессорной и кластерной вычислительных архитектурах, применения

зональной оптимизации, а также спиральных обобщенных координат для квадратной решетки. Этот эксперимент заключался в моделировании игры "Жизнь".

Рассмотрим методику проведения вычислительных экспериментов с помощью предложенного инструментального средства на примере игры "Жизнь".

После запуска среды выполнения и создания в ней нового окна для проводимого эксперимента, в дереве компонентов необходимо выбрать набор компонентов (решетку, хранилище данных, метрику и правила), которые будут использоваться в эксперименте.

В случае отсутствия требуемых компонентов, их необходимо создать. Опишем процедуру выбора и создания компонентов.

1. Выбор решетки. Как известно из описания игры "Жизнь", для ее моделирования требуется квадратная решетка. Компонент, реализующий функциональность такой решетки, имеется в наборе стандартных компонентов и называется "*Square Basic Grid*".
2. Выбор метрики. Будем моделировать игру "Жизнь" в декартовых координатах. Компонент, реализующий функциональность декартовой метрики, существует в наборе стандартных компонентов и называется "*Cartesians*".
3. Выбор хранилища данных. Как известно из описания игры "Жизнь", при ее моделировании состояние клетки автомата описывается переменной булева типа. Таким образом, требуется компонент, реализующий функциональность хранилища булевых данных для декартовой метрики. Он имеется в наборе стандартных компонентов и называется "*Booleans for Cartesians*".
4. Выбор правил. Предположим, что компонент, реализующий правила игры "Жизнь" не входит в набор стандартных компонентов. Разработаем новый компонент и используем его в эксперименте.
   4.1. Разработка компонента. Запустим средство разработки программ на языке программирования *C++* (обычно – *Microsoft Visual C++* 6) и создадим в нем библиотеку, содержащую класс требуемого компонента. Он должен быть наследником класса `CAUniRules` библиотеки *CADLib*, так как именно этот базовый класс обеспечивает независимость выполняемого компонента от вычислительной платформы. В разрабатываемом классе необходимо переопределить только функцию переходов `SubCompute`, которая примет вид:

```
DATUM(CACrtsBool2DDatum);                  Доступ к хранилищу данных.
CACell c;                                  Переменная для хранения текущей
                                           клетки.
CACell neig[12];                           Массив для хранения соседей.
int alive=0;                               Переменная для хранения числа
                                           "живых" соседей.
for(CACell i=z.a1; i<=z.b1; i++)           Цикл по всем клеткам.
  for(CACell j=z.a2; j<=z.b2; j++) {
    alive=0;                               Обнуляем число "живых" соседей.
    c=GET_METRICS->ToCell(i,j,0);          Получаем идентификатор текущей
                                           клетки.
    GET_METRICS->GetNeighbours(c,neig);    Получаем идентификаторы всех
                                           соседей.
    for (unsigned int k=0;                 Цикл по всем соседям.
    k<GET_METRICS->GetNeighboursCount();
    k++) if (datum->Get(neig[k]))alive++;  Вычисляем количество "живых"
                                           соседей.
    if (datum->Get(c))                     Если клетка "жива",
        datum->Set(c,alive==2||alive==3);      она выживет при двух или
                                               трех "живых" соседях.
    else                                   Иначе (если клетка "мертва"),
        datum->Set(c,alive==3);                она оживет при трех "живых"
                                               соседях.
}
return true;                               Эксперимент не закончен.
```

За основу компонента правил можно взять один из стандартных. Большую часть кода занимают операции общие для всех правил автомата: перебор клеток в цикле, доступ к соседям и т.п. Реализацию операций такого рода можно заимствовать из многочисленных примеров, которыми сопровождается инструментальное средство или из соответствующих статей автора и вспомогательных файлов. Те строки, которые имеют непосредственное отношение к логике игры "Жизнь" (тело функции переходов), выделены в приведенном фрагменте кода курсивом.

4.2. Компиляция компонента. Компонент, с помощью компилятора, собирается в *DLL*-библиотеку.

4.3. Установка компонента в инструментальном средстве. Для установки, удаления и изучения компонентов в средстве *CAME&L* имеется специальный инструмент, "*Components Manager*". Необходимо установить созданный компонент с его помощью.

4.4. Выбор компонента. Теперь новый компонент появится в дереве компонентов и его следует выбрать.

5. Настройка компонентов. Для проведения описываемого эксперимента необходимо установить значение параметра "*Double*" хранилища данных "*Booleans for Cartesians*" в значение "*true*", выбрав тем самым синхронный клеточный автомат для моделирования, так как игра "Жизнь" должна выполняться именно на синхронном автомате.

Когда все описанные действия выполнены, можно переходить к запуску вычислительного эксперимента. Перед этим пользователь имеет возможность изменить начальные состояния клеток.

На уровне базового класса, в разработанном компоненте правил, имеется параметр "*Platform*", позволяющий выбрать платформу (однопроцессорную, многопроцессорную или кластерную), на которой будет выполняться эксперимент.

В случае если была выбрана кластерная платформа, то достаточно запустить эксперимент только на одной вычислительной машине кластера. Эта машина станет "владельцем эксперимента". Среда выполнения, запущенная на ней, разделит решетку на части, сопоставит идентификаторы всем вычислительным узлам сети и отправит каждому из них фрагмент решетки, с которым тому предстоит работать. Для этого на вычислительных узлах, вовлеченных в эксперимент, должна быть открыта среда выполнения, входящая в состав предложенного инструментального средства.

Главная функция "владельца эксперимента" заключается в том, что он управляет и координирует выполнение шагов остальными узлами и именно на этом вычислительном узле формируется состояние всей решетки в целом.

Если принять за единицу измерения промежуток времени, требуемый для выполнения одной итерации при моделировании игры "Жизнь" для однопроцессорной платформы, то экспериментально было установлено, что на двухпроцессорной вычисления выполняются в 1,9 раза быстрее[2], а на кластерной из четырех вычислительных узлов – в 3,5 раза быстрее[3]. Измерения проводились, используя решетку размером 1001×1001 клетку.

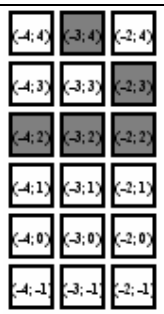
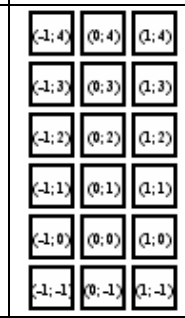
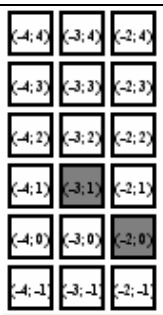
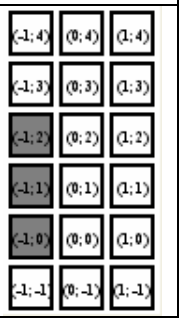
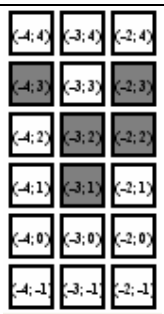
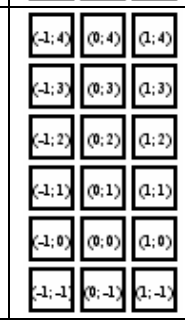
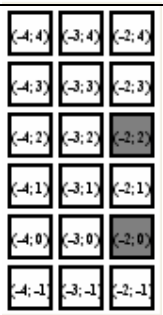
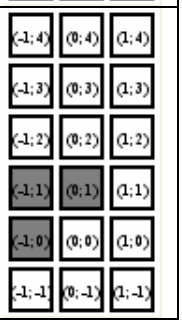
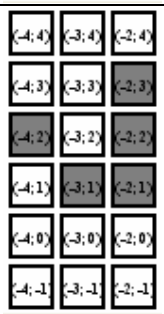
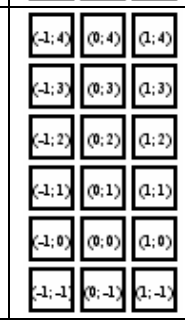
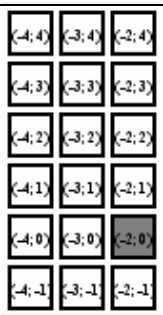
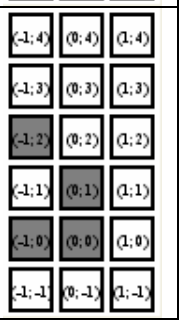
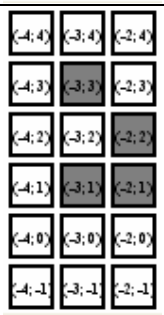
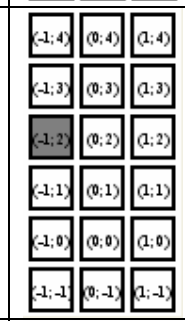
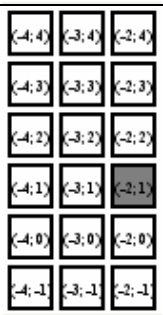
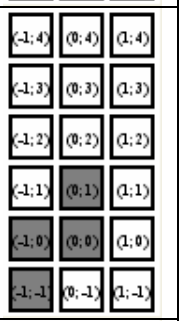
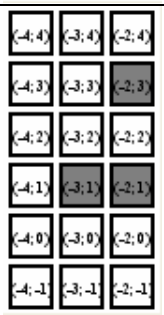
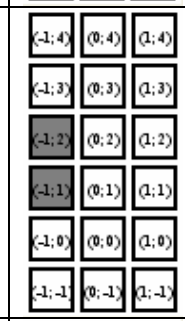
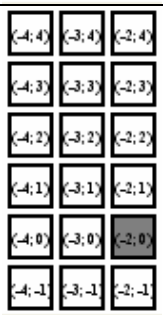
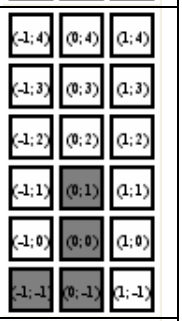
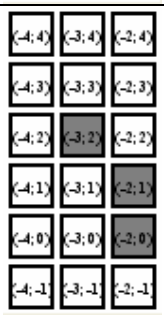
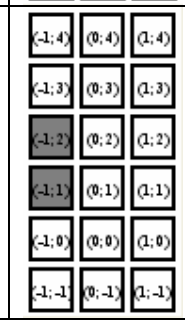
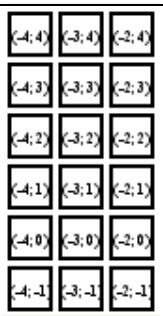
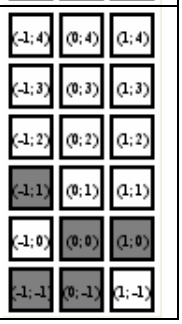
При моделировании игры "Жизнь" на кластерной платформе, при возникновении популяции "планер", можно наблюдать интересный эффект: перемещаясь по решетке, планер будет перелетать с одного вычислительного узла на другой. В таблице 2 показан этот эффект: для двух вычислительных узлов, которые делят решетку так, что клетки с координатами абсцисс

---

[2] Сравнивалось среднее время, необходимое на выполнение итерации при помощи вычислительного узла на базе процессора *AMD Athlon X*2 4800+ и двумя гигабайтами памяти с использованием разработанного компонента правил, работающего в режимах однопроцессорной и многопроцессорной вычислительных архитектур.

[3] Сравнивалось среднее время, необходимое на выполнение итерации на одном и на четырех вычислительных узлах на базе процессоров *Intel Pentium* 4 3400 и одним гигабайтом памяти с использованием разработанного компонента правил, работающего в режимах однопроцессорной и кластерной вычислительных архитектур. При работе в кластере, четыре узла были соединены в квадрат с помощью сети *Ethernet* 100. Задача делилась на четыре части, один из узлов выступал и как "владелец", и как вычислитель.

Таблица 2. Перелет планера с одного вычислительного узла на другой.

Шаг	Узел 1	Узел 2	Шаг	Узел 1	Узел 2
0			6		
1			7		
2			8		
3			9		
4			10		
5			11		

меньшими либо равными минус двум принадлежат первому узлу, а с большими – второму. Приведены фрагменты решеток на первых двенадцати шагах.

Для того чтобы промоделировать игру "Жизнь" в спиральных обобщенных координатах для квадратной решетки, необходимо выбрать другой компонент-метрику (а именно, "*Square Grids Generalized*") и, как следствие, другое хранилище данных ("*Booleans for Generalized*"). Изменений в правила вносить не потребуется.

Необходимо отметить, что при разработке правил пользователю предоставляется существенно большие возможности, чем описание только функции переходов. В приведенном выше теле функции `SubCompute` имеются многочисленные общие операции. Они доступны для определения, так как компонент правил описывает весь эксперимент в целом, включая такие вопросы, как оптимизация вычислений, схема распараллеливания и т.п. Следовательно, общие операции также должны быть доступны разработчику. В результате компонент правил может, например, представлять собой синтаксический анализатор описаний функций переходов на неком языке программирования высокого уровня. Тем самым один компонент правил будет реализовывать множество функций переходов, а не одну.

Следующий раздел этой главы посвящен проведению исследования с помощью инструментального средства *CAME&L*, которое позволило провести классификацию структур, порождаемых одномерными двоичными клеточными автоматами из точечного зародыша.

Все возможные функции переходов одномерного клеточного автомата с окрестностью, состоящей из правой и левой соседних клеток, были пронумерованы целыми числами от 0 до 255. Двоичную запись номера функции можно получить, транспонировав столбец матрицы переходов, содержащий результирующие состояния. Для точечного зародыша были построены все 256 возможных структур. Далее они были сгруппированы в различные классы, используя разнообразные классы эквивалентности.

Простейшая классификация, получившая название *E*-классификации[4] (от англ. "*Equality*"), представляет собой объединение в один класс идентичных структур. Далее, при *EI*-классификации (от англ. "*Inverse*"), в один класс попадают одинаковые структуры, а также те из них, которые являются инверсией друг друга. При *EM*-классификации (от англ. "*Mirror*"), в одном классе оказываются идентичные структуры и те из них, которые представляют собой зеркальное отображение друг друга. При *EIM*-классификации, в одном классе оказываются одинаковые структуры, а также те из них, которые представляют собой зеркальное отображение или инвертированную версию друг друга. При *E*(*I*+*M*)-классификации, в одном классе оказываются идентичные структуры и те из них, которые представляют собой инвертированное зеркальное отображение друг друга.

Далее все эти классификации были выполнены с точностью до сдвига вверх, вниз, вправо или влево на одну клетку. Так появились *EO*-, *EIO*-, *EMO*-, *EIMO*- и *E*(*I*+*M*)*O*-классификации (от англ. "*Offset*"). Затем было решено проверять принадлежность структуры классу, допуская несовпадения в состояниях от одной до пяти клеток. Так появились *ELn*-, *EILn*-, *EMLn*-, *EIMLn*-, *E*(*I*+*M*)*Ln*-, а также *EOLn*-, *EIOLn*-, *EMOLn*-, *EIMOLn*-, *E*(*I*+*M*)*OLn*-классификации (от англ. "*Lapse*"), где *n* может принимать значения от одного до пяти. Можно считать, что первые построенные варианты классификации относились к *L*0-классификациям.

Показано, что существуют автоматы с клетками с памятью и с клетками без памяти, демонстрирующие одинаковое поведение.

Таким образом, в работе выполнен детальный анализ простейших клеточных автоматов, порождающих сложное поведение (в том числе, самовоспроизведение).

В заключительном разделе главы описан набор средств, разработанный для того, чтобы выполнять вычислительные эксперименты с помощью инструментального средства *CAME&L* на *FPGA*-системах (*Field Programmable Gate Array*). Это расширяет множество вычислительных архитектур, на которых могут быть организованы вычислительные эксперименты, используя метод и инструментальное средство, разработанные в диссертации.

---

[4] Было решено называть варианты классификации по аналогии с вариантами классификаций булевых функций.

Идея использования *FPGA*-процессоров для выполнения вычислений при помощи клеточных автоматов не нова. Применение этих систем позволяет существенно повысить эффективность вычислений. При использовании *FPGA*-процессоров, вычисления могут производиться в тысячи раз быстрее, чем на однопроцессорном персональном компьютере. Так, например, Университет Эдинбурга (Шотландия) недавно создал и использует мощнейший суперкомпьютер "*Maxwell*", состоящий из вычислительных узлов, на основе *FPGA*-процессоров, отличающийся чрезвычайно низким, для подобного компьютера, энергопотреблением.

Разработанный комплекс средств позволяет, на основе имеющегося исходного кода компонента правил, автоматически создать управляющий компонент правил, содержащий, в основном, коммуникационные функции, и модуль, выполнимый на *FPGA*-процессоре, который реализует вычислительное ядро эксперимента. При запуске эксперимента с использованием сгенерированного управляющего компонента, модуль загружается в *FPGA*-систему и начинает выполнять вычислительный эксперимент под контролем управляющего модуля. Имеется возможность по завершении любой итерации загрузить состояние решетки клеточного автомата из *FPGA*-процессора в среду выполнения. Для пользователя такой эксперимент неотличим от выполняемого на однопроцессорной, многопроцессорной или кластерной архитектуре.

Таким образом, инструментальное средство *CAME&L* обеспечивает автоматизацию проектирования программного обеспечения *FPGA*-систем, а методы, изложенные в настоящей работе, имеют практическую ценность для автоматизации проектирования различных устройств и вычислительных систем.

Заключение

В диссертации получены следующие результаты:

1. Введено понятие "обобщенных координат" для решеток клеточных автоматов. Разработан метод введения таких координат.
2. На основе метода введения обобщенных координат разработано инструментальное средство автоматизации проектирования программного обеспечения вычислительных экспериментов с использованием клеточных автоматов *CAME&L*.
3. Метод и инструментальное средство *CAME&L* апробированы при автоматизации проектирования программного обеспечения ряда вычислительных экспериментов с использованием клеточных автоматов на однопроцессорной, многопроцессорной, кластерной и *FPGA* вычислительных системах. Ранее было невозможно выполнить это с помощью одного и того же инструментального средства.
4. Проведена классификация всех возможных структур, порождаемых одномерными клеточными автоматами из точечного зародыша. Построено множество разнообразных классов эквивалентности. Показано, что автоматы с клетками с памятью и автоматы с клетками без памяти могут демонстрировать одинаковое поведение. Таким образом, выполнен анализ простейших клеточных автоматов, порождающих сложное поведение (в том числе, самовоспроизведение).
5. Результаты были внедрены в учебном процессе и при проведении научных исследований в ряде университетов.

На основе выполненной работы создан информационный ресурс "*CAMEL Laboratory*" – http://camellab.spb.ru, который содержит инструментальное средство *CAME&L*, доступное для свободного использования вместе с множеством разработанных компонентов и примеров его применения.

Список публикаций

1. *Наумов Л.* Как увеличить скорость "Жизни", или Эффективная организация данных для повышения скорости поиска клеток и разрешения отношений соседства при реализации клеточного автомата Джона Хортона Конвея "Жизнь". Части I и II // Информатика. – 2001. – № 33 (322) – С. 25–27; – № 34 (323) – С. 20–24; Компьютеры + Программы. – 2001. – № 10 – С. 68–73.

2. *Naumov L.* Generalized Coordinates for Cellular Automata Grids / Computational Science – ICCS 2003. – Springer-Verlag. – 2003. Part 2 – pp. 869–878.
3. *Наумов Л., Шалыто А.* Клеточные автоматы. Реализация и эксперименты // Мир ПК. – 2003. – № 8 – С. 71–78.
4. *Наумов Л.* CAME&L – среда моделирования и библиотека разработчика клеточных автоматов / Труды XI Всероссийской научно-методической конференции Телематика'2004. – СПб.: СПбГУ ИТМО. – 2004. – Том 1 – С. 184–186.
5. *Наумов Л.* CTP (Commands Transfer Protocol) – Сетевой протокол для высокопроизводительных вычислений / Труды XI Всероссийской научно-методической конференции Телематика'2004. – СПб.: СПбГУ ИТМО. – 2004. Том 1 – С. 188–189.
6. *Наумов Л., Шалыто А.* "Цветные" клеточные автоматы // Мир ПК. – 2004. – № 5 – С. 64–71.
7. *Naumov L.* CAME&L – Cellular Automata Modeling Environment & Library / Cellular Automata. Sixth International Conference on Cellular Automata for Research and Industry, ACRI-2004. – Springer–Verlag. – 2004 – pp. 735–744.
8. *Наумов Л.* CTP (Commands Transfer Protocol) v. 1.2 – Новая версия сетевого протокола для высокопроизводительных вычислений / Труды XII Всероссийской научно-методической конференции Телематика'2005. – СПб.: СПбГУ ИТМО. – 2005. Том 1 – С. 92–93.
9. *Наумов Л.* Преимущества использования обобщённых координат для многомерных решёток клеточных автоматов / Труды XII Всероссийской научно-методической конференции Телематика'2005. – СПб.: СПбГУ ИТМО. – 2005. Том 1 – С. 93–95.
10. *Наумов Л.* CAME&L – средство для осуществления параллельных и распределенных вычисления на основе клеточных автоматов / Технологии распределенных вычислений. – 2005 – С. 284–286.
11. *Наумов Л.* Решение задач с помощью клеточных автоматов посредством программного обеспечения CAME&L (Части I и II) // Информационно-управляющие системы. – 2005. – № 5 – С. 22–30; – № 6 – С. 30–38.
12. *Наумов Л., Шалыто А.* Классификация структур, порождаемых одномерными двоичными клеточными автоматами из точечного зародыша // Известия РАН. Теория и системы управления. – 2005. – № 5 – С. 137–145. Журнал из списка ВАК.
13. *Наумов Л.* Сравнение специализированных сетевых протоколов CTP (Commands Transfer Protocol) и IL (Internet Link) / Труды XIII Всероссийской научно-методической конференции Телематика'2006. – СПб.: СПбГУ ИТМО. – 2006. Том 1 – С. 266–267.
14. *Наумов Л.* Обзор программного обеспечения для решения задач с использованием клеточных автоматов // Телекоммуникации и информатизация образования. – 2006. – № 2 – С. 114–125.

Тиражирование и брошюровка выполнены в
центре "Университетские телекоммуникации",
Санкт-Петербург, Кронверкский пр., д. 49; тел.: (812) 233-46-69
Тираж 100 экз.