ПРОБЛЕМА ОПИСАНИЯ И СИНТЕЗА РАСПРЕДЕЛЕННЫХ ИМИТАЦИОННЫХ МОДЕЛЕЙ СЛОЖНЫХ МНОГОКОМПОНЕНТНЫХ СИСТЕМ

Ю.И. Бродский (Москва)

Введение, постановка задач

Работа посвящена проблемам имитационного моделирования достаточно широкого класса сложных систем, который характеризуется атомистическим и фрактальным устройством своих представителей. Такое понимание сложной системы отражено, например, в одноименной статье Н.П. Бусленко в БСЭ [17]: «Сложная система – составной объект, части которого можно рассматривать как системы, закономерно объединённые в единое целое в соответствии с определенными принципами или связанные между собой заданными отношениями» – т.е., компоненты сложной системы сами могут быть сложными системами. При этом считается, что хорошо известно устройство «атомов» системы, их поведение, а также закономерности их взаимодействия между собой. Задачей имитационного моделирования в данном случае является воспроизведение поведения всей системы в целом, с целью выяснения ее возможностей, особенностей, возможно решению относительно нее каких-либо управленческих или оптимизационных задач.

Эта задача далеко не тривиальна. Достаточно сложной является даже входящая в нее задача описания всего, что известно о сложной системе. Выше было сказано о наличие знания об устройстве «атомов» системы и их взаимосвязях. Однако сложность здесь в том, что такое знание обычно плохо структурировано: мы редко знаем о какой-либо компоненте моделируемой системы ровно столько, сколько необходимо и достаточно для построения имитационной модели. Если мы знаем меньше – хорошо бы понимать, что еще необходимо узнавать, или от каких особенностей будущей модели можно отказаться. Если мы наоборот, знаем что-то сверх необходимого относительно какой-либо составляющей системы, – хорошо бы уметь исключать такое знание из формального описания, так как подобных составляющих могут быть тысячи, и их избыточные описания, скорее всего, осложнят в дальнейшем проектирование модели.

Тем более сложной является компьютерная реализация имитационной модели сложной системы. Хотелось бы, чтобы проектирование и реализация модели осуществлялись максимально автоматизировано, на основании имеющегося формального описания сложной системы. Желательно также максимально исключать из проекта реализации модели наиболее трудоемкое императивное программирование.

Одним из магистральных направлений развития современной информатики являются распределенные и параллельные вычисления. В связи с этим хотелось бы получать имитационные модели сложных систем, ориентированные на распределенные и высокопроизводительные вычислительные системы.

Обзор существующих решений, уточнение задач

В работе [9] был дан обзор ряда существующих решений в области описания, проектирования и реализации имитационных моделей сложных многокомпонентных систем, кратко перечислим их в порядке появления.

Система GPSS (General Purpose Simulation System), 1961 г., Дж. Гордон, IBM.

Агентное моделирование и программирование, 2-я половина 70-х – 80-е гг., C. Hewitt [2], Y. Shoham [3, 4], системы ABMS (Agent-Based Modeling and Simulation), с конца 90-х.

Инструментальная система MISS (Multilingual Instrumental Simulation System),

1990 г., ВЦ РАН [12].

Унифицированный язык моделирования UML (Unified Modeling Language), середина 90-х, Г. Буч, Д. Рамбо, И. Якобсон [18]; OMG, UML Partners.

Спецификация HLA (High Level Architecture), 2-я половина 90-х, DMSO (Defence Modelling & Simulation Office) Министерства Обороны США.

Инструментальная система моделирования AnyLogic 2000 г., AnyLogic (ранее XJ Technologies) [5, 21].

В наибольшей степени решает поставленные во введении задачи система GPSS, видимо это и позволяет ей сохранять приверженцев на протяжении более полувека. Однако ее предметная область существенно уже, чем заявленная во введении – это различные системы, укладывающиеся в парадигму процессов массового обслуживания.

В основе синтеза многокомпонентной системы лежит идея, высказанная в работе Н.П. Бусленко[18]: дать каждой из компонент, про которые нам все известно, максимально проявить себя, учитывая при этом и все межкомпонентные связи. Это обусловило появление с середины 70-х идей агентного программирования и моделирования, а затем и систем ABMS. Однако на предлагаемых ими агентах лежит некий отпечаток антропоморфности – об их поведении говорится в терминах «убеждений», «обязанностей», «способностей» и т.д. Возникает вопрос – а каким должен быть универсальный агент и его поведение, достаточные для моделирования сложных многокомпонентных систем и в то же время максимально простые, близкие к необходимым условиям такого моделирования?

Про инструментальную систему MISS [12] можно сказать, что это была первая действующая система модельно-ориентированного программирования – такого, где основной единицей программы является модель, помимо характеристик и методов, наделенная законченным поведением – умением стандартно отвечать на стандартные запросы. В то же время, этой системе не хватает теоретического обоснования – например, многие вычисления в ней параллелятся естественным образом, но остается неясным, можно ли этим пользоваться, не приведет ли это к конфликтам за ресурсы.

Язык UML предназначен в первую очередь для проектирования сложных программных систем, хотя может быть применен и для описания сложных систем. Основные его недостатки – избыточность и тяжеловесность, существенно затрудняющие сквозные решения: от описания – до программного кода. Также он оставляет весь проект в рамках объектно-ориентированной императивной парадигмы, которая сложна и не всегда адекватна задачам имитационного моделирования [6, 7].

Спецификация HLA предназначена для создания программного обеспечения, позволяющего объединять в распределенной системе готовые имитационные модели, часть из которых может быть аналоговой – реализованной «в железе».

Система AnyLogic [2, 21] – по-видимому, наиболее мощное решение из имеющихся в настоящее время на рынке инструментальных средств имитационного моделирования. Помимо рассматриваемого в данной работе агентного моделирования она также дает средства моделирования системной динамики и процессов массового обслуживания (отсюда и название AnyLogic). Оставаясь внутри объектно-ориентированной парадигмы, система дает средства заметного упрощения процесса программирования, разделения логики поведения агентов и их функциональности. Тем не менее, также возникает ряд теоретических вопросов, например, до каких пределов можно довести такое упрощение и для каких классов имитационных моделей.

Основной вывод обзора – ни одна из попавших в него систем не решает полностью все поставленные во введении задачи. Кроме того, все эти системы дают более или менее мощные средства решения некоторых из этих задач, но не дают

теоретических обоснований, почему предлагаются именно эти средства, какие средства необходимы и достаточны для построения каких классов имитационных моделей.

Гипотеза о замкнутости и ее следствия

Из работ Н.П. Бусленко (например, [16, 17]), мы знаем, что если имитационная модель реализуется на компьютере (а если она достаточно сложна, то только так ее и можно реализовать), то ее траектория будет из класса кусочно-линейных агрегатов, просто потому что количество вычислений должно быть конечным.

Гипотеза о замкнутости есть предположение о том, что наших знаний о сложной системе достаточно для того, чтобы построить ее имитационную модель, т.е., исходя из начальных значений характеристик, воспроизвести их динамику на отрезке $[0,T]$. Таким образом, задача построения имитационной модели всегда есть задача создания «демона Лапласа» [19, 20].

Будем называть модель замкнутой в точке $t \in [0,T)$, если найдется число $\Delta t > 0$, $t + \Delta t \in (0,T]$, которое будем называть *отрезком прогноза модели для точки* t, такое что: 1). На основании характеристик модели $\vec{X}(t)$ можно определить, есть ли в точке t разрыв траектории первого рода $\Delta\vec{X}(t)$, и если он есть – вычислить его. 2). Далее, на интервале $(t, t+\Delta t)$, траектория модели, выходящая из точки $\vec{X}(t)+\Delta\vec{X}(t)$, является непрерывной функцией времени и однозначно вычисляется по начальным значениям характеристик модели $\vec{X}(t)$.

По-видимому, требование замкнутости модели в любой точке $t \in [0,T)$ есть необходимое условие построения модели – непонятно, как ее строить, если есть точки, из которых невозможен даже малый шаг вперед.

В работах [1, 7] показано, что достаточным это условие не является. Там же показано, что локальная замкнутость становится достаточным условием построения модели, если дополнить ее требованием непрерывности траектории модели слева (обусловленности любого состояния модели некоторой его предысторией). Это – некий аналог теоремы существования для имитационной модели. Весьма важные вопросы единственности, устойчивости и зависимости от начальных значений характеристик, по-видимому, должны выясняться в ходе имитационных экспериментов с моделью.

Весьма важны (быть может, даже важнее теоремы существования) следствия гипотезы о замкнутости – функциональная (следовательно, однозначная) зависимость скачка траектории, ее последующей непрерывной эволюции и размера отрезка прогноза, от значений характеристик модели в начальной точке. Во-первых, функциональные зависимости естественно реализовывать в функциональной парадигме программирования; во-вторых, если удается эквивалентно распараллелить данные вычисления, они должны быть бесконфликтны – конфликт всегда есть нарушение однозначности вычислений; в третьих, необходимость на каждом шаге моделирования вычислять возможный разрыв, непрерывную эволюцию траектории и окончание отрезка прогноза, определяет облик предлагаемого ниже универсального агента.

Модельный синтез и модельно-ориентированное программирование

Модельный синтез и модельно-ориентированное программирование, как методы описания, синтеза и программной реализации имитационных моделей сложных многокомпонентных систем, развивались в отделе Имитационных систем ВЦ АН СССР и затем ВЦ РАН, с конца 80-х гг. Основные их идеи изложены в работах [1, 6-12], а сами термины впервые введены в работе [7]. В основе модельного синтеза лежит понятие модели-компоненты – универсального агента. Модель-компонента подобна

объекту объектного анализа, но снабженному не только характеристиками и способными делать что-то полезное, если их вызовут, методами, а неким аналогом системных служб операционной системы, всегда функционирующим и готовым давать стандартные ответы на стандартные запросы внутренней и внешней среды модели.

Предлагается, основываясь на гипотезе о замкнутости и ее следствиях, формализовать семейство имитационных моделей сложных систем семейством родов структур [6, 7] в смысле Н. Бурбаки [15]. Базисными множествами представителей семейства являются совокупности множеств характеристик модели, методов (того, что модель умеет делать) и событий (того, на что модель должна уметь реагировать). Семейство родов структур «модель-компонента» обладает двумя важными свойствами:

1. Организация имитационных вычислений однотипна для всех представителей семейства. Притом, значительная часть этих вычислений может выполняться параллельно. Это означает возможность создания универсальной программы, ориентированной на высокопроизводительные или распределенные вычисления, способной запустить на выполнение любую имитационную модель, если та является математическим объектом, снабженным структурой рода семейства «модель-компонента».
2. Семейство родов структур «модель-компонента» оказывается замкнутым, относительно операции объединения моделей-компонент в модель-комплекс. Комплекс, полученный объединением моделей-компонент, сам принадлежит семейству родов структур «модель-компонента», и, следовательно, может включаться в новые комплексы, а организация процесса его имитационных вычислений может осуществляться той же самой универсальной программой.

Приведенные выше свойства семейства родов структур «модель-компонента» позволяют предложить новый модельно-ориентированный метод программирования для программной реализации имитационных моделей сложных систем.

Программный комплекс при этом подходе видится как комплекс моделей-компонент, чье поведение нет необходимости специально организовывать (например, вызывая какие-либо методы) – все компоненты всегда ведут себя так, как умеют. Программирование состоит в описании устройства и поведения компонент (по сути – в описании соответствующего рода структуры) и в описании построения комплексов из компонент. Для подобных описаний предлагается декларативный язык ЯОКК (язык описания комплексов и компонент), подробно описанный в работах [7, 9, 11]. Описатели ЯОКК компилируются не в машинный код, а в базу данных модели (что снимает вопрос о качестве компиляции – остается лишь вопрос о ее правильности). Первым воплощением идеи модельно-ориентированного программирования можно считать систему MISS [12], в настоящее время в ВЦ РАН ведутся работы по созданию современной версии подобной системы [13, 14].

Выводы

Предложенные концепции модельного синтеза и модельно-ориентированного программирования позволяют полностью решить все сформулированные во введении задачи – формально описать на языке ЯОКК имеющиеся знания об «атомах» сложной многокомпонентной системы и их связях между собой; автоматически по этим описаниям построить синтез модели сложной системы, путем компиляции описателей ЯОКК в базу данных модели; далее остается запрограммировать методы модели (ориентируясь на функциональную парадигму) и заполнить в базе данных начальные значения ее характеристик.

Таким образом, из проекта программной реализации имитационной модели сложной системы удается полностью исключить императивное программирование,

наиболее сложное как в разработке, так и в отладке, а получаемый код оказывается ориентированным на высокопроизводительные и распределенные вычисления.

Работа выполнена при финансовой поддержке РФФИ в рамках научного проекта №13-01-00499-а.

Литература

1. Brodsky Yu.I. Model synthesis and model-oriented programming – the technology of design and implementation of simulation models of complex multicomponent systems //In the World of Scientific Discoveries, Series B, 2014, Vol. 2, No 1, P. 12-31.
2. Hewitt C. Viewing Control Structures as Patterns of Passing Messages //Journal of Artificial Intelligence. June 1977.
3. Shoham Y. Agent-oriented programming //Artificial Intelligence, vol. 60, 1993. P. 51-92.
4. Shoham Y. MULTIAGENT SYSTEMS: Algorithmic, Game-Theoretic, and Logical Foundations Cambridge: Cambridge University Press, 2010, 532 p.
5. Боев В.Д., Кирик Д.И., Сыпченко Р.П. Компьютерное моделирование: Пособие для курсового и дипломного проектирования. — СПб.: ВАС, 2011. — 348 с.
6. Бродский Ю.И. Роды структур Н. Бурбаки в задаче синтеза имитационных моделей сложных систем и модельно-ориентированное программирование //ЖВМ и МФ, 2015, том 55, № 1, с. 153–164.
7. Бродский Ю.И. Модельный синтез и модельно-ориентированное программирование М.: ВЦ РАН, 2013, 140 с.
8. Бродский Ю.И. Модельный синтез и модельно-ориентированное программирование как технология реализации имитационных моделей сложных многокомпонентных систем, с ориентацией на параллельные и распределенные вычисления //Материалы конференции «Имитационное моделирование. Теория и практика». ИММОД-2013. – Казань: Изд-во «Фэн» Академии наук РТ, 2013. – Т1, С. 114-118.
9. Бродский Ю.И. Распределенное имитационное моделирование сложных систем М.: ВЦ РАН, 2010, 156 с.
10. Бродский Ю.И., Мягков А.Н. Декларативное и императивное программирование в имитационном моделировании сложных многокомпонентных систем //Инженерный журнал: наука и инновации. 2012. № 2 (2). С. 33.
11. Бродский Ю.И., Павловский Ю.Н. Разработка инструментальной системы распределенного имитационного моделирования. //Информационные технологии и вычислительные системы, №4, 2009, С. 9-21.
12. Бродский Ю.И., Лебедев В.Ю. Инструментальная система имитации MISS М.: ВЦ АН СССР, 1991, 180 с.
13. Бродский Ю.И. Программное обеспечение для макета рабочей станции пиринговой сети распределенного имитационного моделирования «Прототип-Д» //Свидетельство №2015613336 о государственной регистрации компьютерной программы от 12.03.2015.
14. Бродский Ю.И. Анализатор типов данных для компилятора с языка описаний комплексов и компонент (ЯОКК) //Свидетельство №2015613334 о государственной регистрации компьютерной программы от 12.03.2015.
15. Бурбаки Н. Теория множеств. М.: Мир. 1965. 456 с.
16. Бусленко Н.П. Моделирование сложных систем М.: Наука, 1978, 400 с.
17. Бусленко Н.П. Сложная система //Статья в Большой Советской Энциклопедии, 3-е изд., М.: Советская энциклопедия, 1969-1978.

18. Буч Г., Рамбо Д., Якобсон И. Введение в UML от создателей языка. 2-е изд.: Пер. с англ. Н. Мухин М.: ДМК Пресс, 2012. 494 с.
19. Лаплас П.С. Изложение системы мира М.: Наука, 1982. 676 с.
20. Лаплас П.С. Опыт философии теории вероятностей Пер. с фр., Изд.2, М.: URSS, 2011. 208 с.
21. Осоргин А.Е. AnyLogic 6. Лабораторный практикум Самара: ПГК, 2011. 100 с.