Е. А. Павлова, А. А. Станкевичус

ИССЛЕДОВАНИЕ СЛОЖНОСТИ ВЕРИФИКАЦИИ МОДЕЛЕЙ ДЛЯ ПРАВИЛ ЛОГИЧЕСКИХ КОМПОНЕНТОВ С УЧЕТОМ СВОЙСТВ БЕЗОПАСНОСТИ

Методы верификации применяются для обеспечения безопасности электронных схем, коммуникационных протоколов и программного обеспечения. Основная сложность, которую приходится преодолевать в ходе верификации при помощи проверки модели, обусловлена эффектом «комбинаторного взрыва» в пространстве состояний проверяемой модели. В случае, когда число состояний достаточно велико, процесс проверки может оказаться неэффективным или невозможным, так как требует значительных затрат машинного времени и памяти. Настоящая работа посвящена оценке эффективности применения методов проверки модели для верификации компонентов с большим числом достижимых состояний на примере правил компьютерных игр в классе пошаговых стратегий (порядка 10^{40} состояний).

Современная индустрия компьютерных игр характеризуется высокими темпами роста (по различным оценкам, 20–25 % в год) и значительным годовым доходом (17,9 млрд долларов для европейского рынка, по оценкам [1]). Существуют компьютерные игры, использующие глобальную сеть Интернет, которые насчитывают миллионы игроков. Многие современные игры имеют развитую бизнес-модель, предполагающую применение в игре электронных денег для расчетов между игроками и оплаты возможностей игры. При некорректных правилах игроки находятся в неравных условиях, что может привести к мошенничеству [2]. Одной из важнейших задач при разработке игр является обеспечение безопасности правил.

Верификация правил игры — один из методов проверки компьютерных игр в классе пошаговых стратегий [3]. В настоящей работе под стратегической игрой понимается компьютерная игра, в которой значительное влияние на исход игры оказывают способности игроков принимать решения и вырабатывать стратегию. Термин «пошаговая» означает, что течение игрового процесса подразделяется на строго определенные части — ходы. Игроку дается некоторое время (иногда неограниченное) на размышление и выполнение хода. В игре может участвовать один или несколько игроков. В качестве исследуемого жанра игр были выбраны пошаговые стратегии, так как именно в этом жанре ключевыми задачами разработки являются создание и верификация логического компонента игры, основанного на правилах игры [3].

В настоящей работе правила игры понимаются как описание сущностей предметной области игры, их допустимого взаимодействия друг с другом, цели игры и условий выигрыша. Игры традиционно верифицируют путем длительного (иногда многолетнего) тестирования [3] программной реализации игры. При таком подходе ошибки, допущенные на этапе проектирования, выявляются уже после реализации в виде программного кода. Для обнаружения ошибок на более ранних этапах разработки возможно применение тестирования на основе моделей (model-based testing) и методов формальной верификации моделей (model checking). Одним из недостатков применения тестирования на основе моделей является высокая сложность вычисления тестов. Цель настоящей работы состоит в формализации часто используемых критериев проверки корректности правил логических компонентов игры и в оценке сложности формальной верификации модели правил на этапе проектирования в соответствии с определенными критериями.

Расчет числа состояний. В настоящей работе рассматривается разработанный ранее метод [4], использующий методы проверки моделей для верификации правил игры. Метод состоит в построении модели правил игры на основе их формального описания на разработанном ранее языке [5], выборе и формализации критерия проверки и верификации модели при помощи инструментального средства (model checker).

Проверяемые свойства моделей чаще всего подразделяют на свойства безопасности (safety properties), свойства живучести (liveness properties) и стабильности (persistence properties) [6]. Свойства безопасности содержат утверждения о том, что нечто нежелательное при любом варианте работы системы никогда не случается [6]. В настоящей работе рассматривается именно этот тип свойств. При верификации безопасности правил на этапе проектирования основное внимание уделяется проверке сбалансированности правил. Это понятие из предметной области компьютерных игр в общем случае означает, что не будет нарушен ни один из критериев справедливости правил [3]. Возможные определения баланса приведены, например, в работе [3].

Многие критерии баланса основаны на оценке состояний игроков в конкретные моменты игры. Состояние игроков может включать положение их фигур на игровом поле и значения некоторых свойств фигур. Таким образом, для проверки правил игры достаточно выделить из описания правил упрощенную модель, содержащую информацию, достаточную для проверки правил по определенным критериям.

Рассмотрим пример описания игры, содержащий такую информацию. Пусть дано описание правил игры (разработанное на этапе проектирования). Игра между двумя игроками происходит на поле размерности m x n, где $m \geq 1$, $n \geq 1$. Клетки поля квадратные, единичного размера. Используется k фигур, где $k \geq 0$. В течение игры на поле может находиться от k (в начале игры) до 0 (по окончании игры) фигур. Начальное значение $k \geq 2$, так как у каждого игрока в начале игры есть хотя бы по одной фигуре. На одной клетке не может одновременно находиться более одной фигуры, поэтому ограничения на размеры поля включают в себя условие: $mn \geq k$. Правила допускают единственное свойство фигуры — «величина повреждения» (hit point) h, где $h \geq 0$ и начальное значение $h \geq 1$. За один ход каждый игрок может передвинуть каждую из своих фигур в произвольном порядке на одну из соседних незанятых клеток. Кроме того, игрок может нанести своей фигурой повреждение фигуре другого игрока, находящейся на соседней клетке. Нанесение повреждения выражается в уменьшении величины повреждения h на единицу.

Проведем оценку возможного числа состояний (позиций) в такой игре. Пусть все фигуры полагаются различными. Оценим число возможных позиций фигур на поле без учета возможности повреждения фигур (число размещений из mn по k):

$$A_{mn}^{k} = \binom{mn}{k} k! = \frac{(mn)!}{(mn-k)!}.$$

Если фигуры имеют свойство «величина повреждений», то возможны h^k наборов фигур, где в каждом наборе для j-й фигуры i-го игрока возможное повреждение $\rho_{ij} \in [1, h]$, i — номер игрока, $i \in [0, 1]$, а j — номер фигуры в списке фигур игрока, $j \in [0, k]$. Итоговое число возможных позиций фигур на поле

$$\sum_{l=1}^{k} A_{mn}^{l} h^{l} = \sum_{l=1}^{k} \frac{(mn)!}{(mn-l)!} h^{l}.$$

Учитывая, что число размещений без повторения меньше, чем число размещений с повторениями, получим

$$A^{k}_{mn} \leq (m,\ n)^{k},$$

$$\sum_{l=1}^{k} \frac{(mn)!}{(mn-l)!} h^{l} \leq \sum_{l=1}^{k} (mn)^{l} h^{l}.$$

Так как $m \geq 1$, $n \geq 1$, $h \geq 1$, $l \geq 1$, то $\forall l \in [1, k]$ $((mnh)^{l})' = l\,(mnh)^{l-1} > 0$, следовательно, функция $f(l) = (mnh)^{l}$ строго монотонно возрастает.

Поэтому для $l \in [1, k]$ $(mnh)^{l} \leq (mnh)^{k}$,

$$\sum_{l=1}^{k} \frac{(mn)!}{(mn-l)!} h^{l} \leq k(mnh)^{k}.$$

Число возможных позиций $g(m,\ n,\ k,\ h) \le k(mnh)^k$.

Возможно уменьшение оценки числа состояний, если считать фигуры с совпадающими характеристиками и принадлежащие одному игроку одинаковыми. В случае логических компонентов с большим числом состояний на модель накладываются ограничения справедливости (fairness constraint), которые позволяют рассматривать лишь те последовательности состояний (пути) в графе состояний, которые удовлетворяют этим ограничениям. Одним из таких ограничений является соответствие путей определенным стратегиям поведения [3], что позволяет исключить из рассмотрения редко встречающиеся на практике «неразумные» [3] варианты поведения игроков.

Сложность верификации. Сложность проверки свойств безопасности зависит не только от числа состояний модели, но и от алгоритма, реализованного в конкретном инструментальном средстве. В настоящей работе применялось широко используемое (по оценкам [6]) при проверке программного обеспечения средство SPIN [7]. Время и память, требуемые для проверки свойств безопасности при помощи SPIN, линейно зависят от числа состояний [7]. С учетом проведенной оценки числа состояний объем памяти, необходимой для проверки свойства сбалансированности при помощи алгоритмов SPIN, полиномиально зависит от каждого из измерений поля и от числа возможных повреждений фигуры. Размер памяти зависит экспоненциально от числа фигур. Время, необходимое для проверки сбалансированности полиномиально зависит от измерений поля и от числа возможных повреждений фигуры. Время зависит экспоненциально от числа фигур.

Оценим зависимость сложности верификации от характеристик проверяемых критериев, формализуемых в виде формул временной логики. Например, правила игры, не допускающие выигрыша одного из игроков, будем считать несправедливыми. Сформулируем свойство «в игру может выиграть i-й игрок» в логике CTL: EF (win=i), где i — переменная, в которой хранится номер игрока, win — переменная, содержащая номер победителя или 0, если игра не окончена. SPIN поддерживает только LTL. Поэтому в SPIN проверим отрицание этого свойства

$\overline{EF(win = i)} = A!(F(win = i))$.

Будем считать несправедливыми правила, при которых i-й игрок всегда проигрывает за первые b ходов. Сформулируем отрицание критерия «Существуют варианты игры, в которых i-й игрок не проигрывает за первые b ходов» и его отрицание, count — счетчик ходов:

$\overline{EF(((!win)! = i)\ \&\ (count \le b))} = A!(F(((!win)! = i)\ \&\ (count \le b)))$.

Будем считать несправедливыми правила, при которых состояние i-го игрока всегда хуже или равно состоянию j-го на протяжении первых b ходов:

$\overline{EF(v_i > v_j)} = A!(F(v_i > v_j))$

где v_i, v_j — оценки состояния игроков на момент конкретного хода.

Будем считать несправедливыми правила, при которых состояние i-го игрока непрерывно ухудшается на протяжении первых b ходов:

$\overline{EF(v_i > 0)} = A!(F(v_i > 0))$.

Несправедливыми будем считать правила, когда отношение ущерба $damage_i$, наносимого отрядом i, к его оценочной стоимости $price_i$ отличается более чем в b раз от отношения $damage_j$, наносимого отрядом j, к его оценочной стоимости $price_j$:

$$AG\left(\frac{damage_i}{price_i} = \frac{damage_j}{price_j} \cdot b\right).$$

Сложность проверки LTL формул в SPIN возрастает в r раз, где r в худшем случае зависит экспоненциально от числа временных операторов, используемых в формуле [7]. Формулы, соответствующие многим критериям справедливости, используют не более двух операторов. На память, по данным работы [7], сложность формулы практически не влияет. Заметим, что в общем случае проверка LTL и CTL формул является PSPACE-полной задачей в зависимости от длины проверяемой формулы [6].

Альтернативой верификации при помощи моделей может служить тестирование на основе моделей. Размер и сложность вычисления тестов при помощи автоматов (один из самых распространенных подходов к тестированию моделей) с использованием W, Wp методов [8] равна по порядку $O(\rho^{N-n+1}n^3)$ (O (ρn^3) для N=n), где N — число состояний реализации, а n — число состояний спецификации. Реализация представляет собой модель системы, а спецификация — модель проверяемого свойства. Сложность D-метода в общем случае тоже экспоненциальная.

Заметим, что все оценки сложности являются асимптотическими, так как различные реализации одного алгоритма могут иметь разную эффективность, а идентичные реализации могут по-разному функционировать в отличающихся окружениях. В настоящей работе была проведена оценка производительности разработанного метода проверки для модели правил тестовой игры (описание игры приведено в работах [4, 5]) по критерию (1) для различных значений параметров игры. Оценка производительности проводилась на ЭВМ с процессором Intel Core 2 Duo с тактовой частотой 2 Гц, снабженной 2 Гб ОЗУ, под управлением операционной системы на базе ядра Linux — Ubuntu 4.2. В соответствии с полученными экспериментальными данными и теоретической зависимостью с учетом ограничений справедливости для $m = 8$, $n = 8$, $k = 8$, $h = 8$ требуемая память составит 8 Гб, а требуемое время — 8,5 часа.

Верификация систем с большим числом состояний является ресурсоемкой задачей. Настоящая работа посвящена исследованию сложности проверки свойств безопасности для таких систем на примере логических компонентов компьютерных игр. В работе были формализованы часто используемые критерии проверки корректности правил в виде выражений логик линейного и ветвящегося времени. Проведена асимптотическая оценка сложности верификации модели правил на этапе проектирования в соответствии с определенными критериями.

Время и память, необходимые для проверки модели экспоненциально зависят от длины входа и экспоненциально — от числа временных операторов, используемых при записи критерия проверки. Прогнозируемые время и объемы памяти, необходимые для проверки современных игр, соответствуют возможностям ЭВМ, применяемым на практике при решении задач верификации. Полученные в работе новые результаты могут быть использованы при разработке методов и инструментальных средств для верификации правил логических компонентов.

СПИСОК ЛИТЕРАТУРЫ:

1. *Androvich M.* Forbes: Europe is least mature videogame market, 2008. http://www.gamesindustry.biz/articles/forbes-europe-is-least-mature-videogame-market.

2. *Bono S., Caselden D., Landau G., Miller C.* Reducing the Attack Surface in Massively Multiplayer Online Role-Playing Games // IEEE Security and Privacy. 2009. Vol. 7. № 3. P. 13–19.

3. *Rollings A., Morris D.* Game Architecture and Design. A New Edition. New Riders Publishing. Indianapolis, 2004. — 960 p.

4. *Pavlova E.* The Formal Approach to Computer Game Rule Development Automation // Proceedings of the Third Spring Young Researchers' Colloquium on Software Engineering (SYRCoSE 2009). Moscow, May 28–29, 2009. — P. 119–122.

5. *Павлова Е. А.* Проектирование формального предметно-ориентированного языка (DSL) для разработки правил компьютерных игр в классе пошаговых стратегий // Вестник компьютерных и информационных технологий. 2009. № 4. Машиностроение. 2009. С. 45–52.

6. *Кулямин В. В.* Методы верификации программного обеспечения // Всероссийский конкурсный отбор обзорно-аналитических статей по приоритетному направлению «Информационно-телекоммуникационные системы». 2008. — 117 с.

7. *Holzmann G. J.* The Model Checker SPIN // IEEE Transaction on Software Engineering. 1997. Vol. 23. № 5. P. 279–295.

8. *Broy M., Jonsson B., Katoen J.-P., Leucker M., Pretschner A. (eds.).* Model Based Testing of Reactive Systems. LNCS 3472. Springer, 2005. — 659 p.