УЧЕНЫЕ ЗАПИСКИ КАЗАНСКОГО УНИВЕРСИТЕТА.
СЕРИЯ ФИЗИКО-МАТЕМАТИЧЕСКИЕ НАУКИ

2024, Т. 166, кн. 4
С. 532–554

ISSN 2541–7746 (Print)
ISSN 2500–2198 (Online)

ОРИГИНАЛЬНАЯ СТАТЬЯ

УДК 531.551.1: 681.5

doi: 10.26907/2541-7746.2024.4.532-554

ФОРМАЛЬНЫЙ ПОДХОД К ПРОСТРАНСТВЕННО-ВРЕМЕННОМУ МОДЕЛИРОВАНИЮ ИГРОВЫХ СИСТЕМ

В.В. Кугуракова

Казанский (Приволжский) федеральный университет, г. Казань, 420008, Россия

Аннотация

Представлен инновационный унифицированный подход к формальному моделированию игровых сущностей и их взаимодействий – FAST-GM (Formal Approach to Spatio-Temporal Game Modeling). Предложенная модель интегрирует темпоральные и вероятностные аспекты, обеспечивая комплексное описание динамики игровых систем. Подход FAST-GM основан на расширенной темпоральной логике и теории вероятностей, что позволяет точно описать сложные игровые механики и их эволюцию во времени. Рассмотрено формальное определение игровых сущностей, их состояний и взаимодействий, а также методы интеграции темпоральных и вероятностных элементов. Особое внимание уделено применению модели для анализа игрового баланса, формальной верификации игровых сценариев и автоматизированной генерации тестовых случаев. Обсуждены масштабируемость и адаптивность модели для различных игровых жанров. Ожидается, что предложенный подход FAST-GM станет значительным шагом вперед в формальном моделировании игровых систем, предоставляя разработчикам мощный инструмент для анализа, верификации и оптимизации игровых механик на различных этапах разработки.

Ключевые слова: формальное моделирование, игровая сущность, взаимодействие в играх, темпоральная логика, вероятностная модель, анализ игрового баланса, верификация игровых сценариев, FAST-GM, game studies, геймдизайн

Введение

В современной индустрии разработки компьютерных игр наблюдается постоянное усложнение игровых систем и механик. Это приводит к возрастающей потребности в более формализованных и структурированных подходах к проектированию, анализу и балансировке игр. Традиционные методы, основанные преимущественно на интуиции и опыте разработчиков, становятся недостаточными для обеспечения качества и сбалансированности сложных игровых систем.

Формальное моделирование игровых сущностей и их взаимодействий представляет собой перспективный подход к решению названных проблем. Однако существующие формальные модели часто ограничены в своей способности охватить всю сложность современных игр, особенно в аспектах, связанных с временной динамикой и вероятностными элементами.

Цель данного исследования – разработка унифицированного подхода к формальному моделированию игровых сущностей и их взаимодействий, который интегрирует темпоральные и вероятностные аспекты. Предлагаемая модель, названная

FAST-GM (Formal Approach to Spatio-Temporal Game Modeling), призвана обеспечить комплексное описание динамики игровых систем, позволяющее более точно анализировать и прогнозировать поведение игровых механик.

Работа развивает наши предыдущие исследования в области формализации и анализа игровых механик [1, 2], а также опирается на разработанный нами подход к созданию универсальной структуры корпуса текстов видеоигр [3].

Основные задачи исследования включают:

- разработку формального аппарата для описания игровых сущностей, учитывающего их атрибуты, состояния и поведение во времени;
- интеграцию вероятностных элементов в модель для отражения неопределенности и случайности, присущих многим игровым механикам;
- создание методологии применения разработанной модели для анализа игрового баланса и формальной верификации игровых сценариев;
- исследование масштабируемости и адаптивности модели для различных жанров игр.

Предлагаемый подход основан на расширенной темпоральной логике и теории вероятностей, что позволяет точно описывать сложные игровые механики и их эволюцию во времени. Особое внимание уделено обеспечению практической применимости модели в реальном процессе разработки игр.

Мы опираемся на принципы системной инженерии, в частности на концепции, представленные в [4]. Это позволяет рассматривать игровые системы как сложные иерархические структуры с четко определенными уровнями абстракции. В основе модели FAST-GM лежит трехуровневая структура абстракции, характерная для системной инженерии.

- Концептуальный уровень: описывает общие понятия и взаимосвязи игровых сущностей.
- Логический уровень: предоставляет формальное представление сущностей и их взаимодействий.
- Физический уровень: описывает конкретную реализацию в игровых движках или системах.

Эта структура позволяет последовательно переходить от общих концепций к конкретным реализациям, обеспечивая целостность и согласованность модели на всех уровнях.

Структура статьи такова: сначала представлен обзор существующих подходов к формальному моделированию в game studies[1] [5], затем подробно описана предлагаемая модель FAST-GM, включая её математический аппарат и методологию применения. Далее рассмотрены примеры использования модели для анализа игрового баланса и верификации сценариев. В заключительной части проведено сравнение FAST-GM с существующими подходами и обсуждены перспективы дальнейших исследований в этой области.

Ожидается, что разработанный подход внесет значительный вклад в развитие методов формального анализа и проектирования игровых систем, способствуя повышению качества и сбалансированности современных компьютерных игр.

[1] Game studies (исследования игр) – это междисциплинарная область академических исследований, фокусирующаяся на играх, в частности видеоиграх, их дизайне, игроках и их роли в обществе и культуре. Эта область объединяет подходы из различных дисциплин, включая компьютерные науки, культурологию, психологию, социологию, антропологию и медиаисследования.

1. Связанные работы

В области формального моделирования игровых систем существует ряд значимых исследований, которые служат основой для представленной работы.

1.1. Модель реальных и фиктивных элементов. В монографии [6] предложена общая модель игр – её можно назвать моделью реальных и фиктивных элементов. Она менее детализирована, чем наша модель, но представляет собой важный шаг в формализации игровых систем. Эта модель фокусируется на взаимосвязи между правилами игры и фиктивным миром, что частично отражено в нашем подходе к моделированию игровых сущностей:

$$Game = (Rules, Fiction, PlayerInteraction),$$

где $Rules$ – правила игры, $Fiction$ – фиктивный мир игры, $PlayerInteraction$ – взаимодействие игрока с игрой.

1.2. Архитектура интерактивной драмы. В статье [7] разработаны формальные модели для интерактивных драматических сценариев в рамках построения своей системы Facade. Такие сценарии можно рассматривать как специфический случай игрового моделирования:

$$Scene = (Beats, Characters, DramaticArc),$$

где $Beats$ – ключевые моменты сцены, $Characters$ – персонажи, участвующие в сцене, $DramaticArc$ – драматическая структура сцены.

1.3. Формальная игровая грамматика. В [8] предложена формальная грамматика для описания игровых механик, что близко к нашему подходу. Этот подход больше похож на язык программирования для игр:

$$Mechanic ::= Action|Rule|Resource,$$

где $Action$ – действие в игре, $Rule$ – правило игры, $Resource$ – ресурс в игре.

1.4. Онтология игровых механик. В работе [9] в рамках Game Ontology Project разработана общая онтология для описания элементов игрового дизайна, что можно рассматривать как попытку обобщения игровых элементов:

$$Interface \leftarrow Presentation \leftarrow Rules \leftarrow Goals \leftarrow Entities,$$

где $Interface$ – интерфейс игры, $Presentation$ – представление игровых элементов, $Rules$ – правила игры, $Goals$ – цели игры, $Entities$ – сущности в игре.

1.5. Химия игрового дизайна. В [10] предложена система атомарных элементов геймплея, которая хотя и менее формальна, но стремится к обобщению игровых механик:

$$Game = Atom_1 + Atom_2 + ... + Atom_N, \text{ где } Atom_i = (Action, Simulation, Feedback),$$

где $Atom_i$ – атомарный элемент геймплея, $Action$ – действие игрока, $Simulation$ – симуляция результата действия, $Feedback$ – обратная связь игроку.

1.6. Паттерны игрового дизайна. В книге [11] предложена система паттернов игрового дизайна, которая может рассматриваться как попытка формализации и обобщения игровых элементов. Эта работа послужила источником вдохновения для разработки нами подхода к моделированию повторяющихся элементов в различных жанрах игр:

$$Pattern = (Name, Description, Consequences, Using),$$

где $Name$ – название паттерна, $Description$ – описание паттерна, $Consequences$ – последствия применения паттерна, $Using$ – примеры использования паттерна.

1.7. Генеративный анализ Петри-сетей. В статье [12] использованы формальные методы для генерации игрового контента посредством интегрирования логических и вероятностных элементов и применения подходов, близких к нашему:

$$Content = GenerateStructure(Logic) \times ApplyProbabilities(Structure),$$

где $Content$ – сгенерированный игровой контент, $GenerateStructure$ – функция генерации структуры контента, $Logic$ – логические правила генерации, $ApplyProbabilities$ – функция применения вероятностных элементов, $Structure$ – структура контента, $\times$ – оператор композиции или последовательного применения функций. Сначала применяется функция $GenerateStructure$ для создания базовой структуры контента на основе логических правил, а затем к полученной структуре применяется функция $ApplyProbabilities$ для добавления вероятностных элементов. Таким образом, $\times$ здесь обозначает последовательность операций в процессе генерации игрового контента.

1.8. Формальные модели в гейм-дизайне. С.М. Грюнфогелем в [13] была рассмотрена роль формальных моделей в дизайне компьютерных игр. Эта работа важна для нашего исследования, так как она закладывает основу для применения формальных методов в game studies и разработке игр. С.М. Грюнфогель предложил использовать различные математические подходы, включая теорию игр, теорию автоматов и теорию динамических систем, для моделирования игровых механик. Его подход подчеркивает необходимость гибкости формальных моделей для охвата разнообразия игровых жанров и механик:

$$Game = FormalModel(Mechanics, Dynamics, Aesthetics),$$

где $Mechanics$ – правила и системы игры, $Dynamics$ – поведение игры во время выполнения, $Aesthetics$ – эмоциональные реакции, вызываемые у игрока.

1.9. Предлагаемый формальный подход к пространственно-временному игровому моделированию. Наше исследование развивает и дополняет существующие подходы, предлагая более детализированную и универсальную модель. Мы представляем унифицированный подход к формальному моделированию игровых сущностей и их взаимодействий, который интегрирует темпоральные и вероятностные элементы, а также учитывает специфику различных игровых жанров. Этот подход, названный нами Formal Approach to Spatio-Temporal Game Modeling (FAST-GM), или «Формальный подход к пространственно-временному игровому моделированию», представляет собой комплексное решение для анализа и разработки современных игровых систем.

FAST-GM стремится преодолеть ограничения существующих подходов, предоставляя более полный и гибкий инструментарий для моделирования игровых систем. Детальное описание структуры и компонентов модели представлено в следующем разделе.

FAST-GM отличается (см. табл. 1) от других рассмотренных методологий по нескольким ключевым особенностям:

- высокий уровень абстракции, позволяющий моделировать сложные игровые системы;
- интеграция темпоральных и вероятностных аспектов в единую модель;
- адаптивность к различным игровым жанрам;
- ориентация на практическое применение в анализе и разработке игр.

Таким образом, FAST-GM – это комплексный подход к моделированию игровых систем, объединяющий сильные стороны существующих методологий и расширяющий их возможности для решения современных задач игровой разработки.

2. Формальная модель игровых сущностей

При разработке формальной модели игровых сущностей мы опирались на опыт создания онлайн-инструмента для балансировки видеоигр [14], что позволило учесть практические аспекты применения формальных методов в игровом дизайне (в свою очередь, подход FAST-GM, предложенный в настоящей работе, послужит основой для усовершенствования этого онлайн-инструмента, что позволит более точно и эффективно анализировать и балансировать игровые механики).

2.1. Определение базовых игровых сущностей. Базовые игровые сущности можно классифицировать следующим образом [9]:

A. Активные сущности:
   a) игровые персонажи (управляемые игроком и NPC),
   b) интерактивные объекты (двери, рычаги, транспортные средства),
   c) искусственный интеллект (боты, системы управления).

B. Пассивные сущности:
   a) статические объекты окружения (здания, ландшафт),
   b) ресурсы (здоровье, энергия, валюта),
   c) информационные элементы (задания, журналы, карты).

C. Системные сущности:
   a) игровые правила и механики,
   b) системы физики и коллизий,
   c) системы загрузки и сохранения.

D. Мета-сущности:
   a) игровые сессии,
   b) профили игроков,
   c) достижения и статистика.

Табл. 1

Сравнение подходов к моделированию игровых систем

Подход	Уровень абстракции	Темпоральные аспекты	Вероятностные элементы	Вычислительная сложность	Практическая применимость
2.1	Концептуальный	Не представлены	Не включены	Низкая	Высокая для теоретического анализа
2.2	Логический	Представлены в контексте нарратива	Частично в диалоговых системах	Средняя	Средняя, специфична для нарративных игр
2.3	Логический	Не представлены явно	Не включены	Средняя	Высокая для формального описания механик
2.4	Концептуальный	Не представлены явно	Не включены	Низкая	Высокая для классификации элементов игрового дизайна
2.5	Концептуальный	Частично в механике обратной связи	Частично в механике обратной связи	Низкая	Средняя, фокус на творческом аспекте дизайна
2.6	Логический	Частично в некоторых паттернах	Не включены явно	Низкая	Высокая для анализа и дизайна игровых механик
2.7	Логический	Частично в анализе Петри-сетей	Интегрированы	Высокая	Средняя, фокус на генерации контента
2.8	Концептуальный, Логический	Рассматриваются в контексте динамических систем	Частично рассматриваются	Средняя	Средняя, теоретический фокус
2.9	Концептуальный, Логический, Физический	Интегрированы с использованием темпоральной логики	Являются неотъемлемой частью модели	Высокая (PSPACE-полная для LTL)	Высокая, с ограничениями для крупномасштабных систем

2.1.1. Уровни абстракции игровых сущностей. На концептуальном уровне игровая сущность определяется как абстрактный объект, обладающий определенными характеристиками и способный взаимодействовать с другими сущностями.

На логическом уровне мы формализуем сущность E (сокр. от англ. *Entity*) как фундаментальный элемент игровой системы, обладающий уникальными характеристиками, состоянием и поведением, который формально можно определить как

$$E = \langle I, A, S, T, B \rangle,$$

где I ($Identifier$) – уникальный идентификатор, A ($Attributes$) – множество атрибутов, S ($State$) – текущее состояние, T ($TemporalProperties$) – темпоральные свойства, B ($Behavior$) – общее вероятностное поведение, причём каждый атрибут $a \in A$ представляет собой характеристику сущности, т. е. $A = a_1, a_2, ..., a_n$, где n – количество атрибутов сущности.

На физическом уровне сущность представляется как конкретная структура данных или объект в игровом движке.

Выбор абстракций для представления сущностей в FAST-GM обусловлен следующими факторами:

- универсальность – модель должна быть применима к широкому спектру игровых жанров и стилей;
- гранулярность – уровень детализации должен быть достаточным для точного моделирования, но не избыточным для эффективного анализа;
- расширяемость – возможность легко добавлять новые типы сущностей и атрибуты без изменения базовой структуры модели;
- совместимость – способность интегрироваться с существующими игровыми движками и инструментами разработки.

Листинг 1

Пример формального определения игрового персонажа в FAST-GM

```
Character = (I: ‘‘player_001’’, A: {health_max, strength, speed},
S: [будет детально определено позже],
T: {health_regen: 1/s, ability_cooldown: {fireball: 30s}},
B: {crit_chance: 0.1, dodge_probability: 0.05})
```

2.2. Формальное представление состояний сущностей. Формально определим состояние игровой сущности как кортеж

$$S = \langle V, C, \tau, P \rangle,$$

где V – функция, сопоставляющая атрибутам из множества A их текущие значения ($Values$), C – множество ограничений на значения атрибутов ($Constraint$), τ – временная метка ($time$), P – вероятностные характеристики текущего состояния ($Probability$).

2.2.1. Значения атрибутов. V определяется как функция, отображающая атрибуты в их текущие значения:

$$V : A \rightarrow W,$$

где A – множество атрибутов, W – объединение всех множеств допустимых значений для всех атрибутов.

Для каждого атрибута $a \in A$, $V(a) = w$, где $w \in Da$ и Da – множество допустимых значений для атрибута a.

Da представляет собой все возможные значения, которые может принимать конкретный атрибут a, например:

- для числового атрибута 'здоровье' Da может быть определено как $Da = x \in \mathbb{N} \mid 0 \leqslant x \leqslant 100$, что означает, что здоровье может принимать любое целое значение от 0 до 100;
- для атрибута 'имя персонажа' Da может быть множеством всех допустимых строк определенной длины, например, $Da = s \mid s$ – строка, $1 \leqslant$ длина$(s) \leqslant 20$;
- для атрибута 'класс персонажа' в ролевой игре Da может быть конечным множеством предопределенных значений, например, $Da =$ 'воин', 'маг', 'лучник', 'целитель';
- для атрибута 'координаты' в двумерном пространстве Da может быть определено как $Da = (x, y) \mid x, y \in \mathbb{R},\ 0 \leqslant x \leqslant 1000,\ 0 \leqslant y \leqslant 1000$, где x и y – действительные числа в пределах игрового поля размером 1000×1000.

2.2.2. Ограничения. Множество C содержит ограничения на значения атрибутов $C = c_1, c_2, ..., c_m$, где каждое c_i представляет собой предикат над значениями атрибутов.

2.2.3. Временная метка. Величина τ представляет момент времени, к которому относится данное состояние. Это позволяет отслеживать эволюцию состояния сущности во времени.

Рассмотрим сущность «игрок» (player) в ролевой игре, определив её компоненты как $E_{player} =< I, A, S, T, B >$, где
$I =$ "player_001",
$A = \{$здоровье, сила, опыт$\}$,
$T = \{$скорость_регенерации : 1 ед/мин, длительность_баффов : 5 минут$\}$,
$B = \{$вероятность_крит_удара : 0.1, шанс_уклонения : 0.05$\}$.

Состояние игрока S в конкретный момент времени может быть описано как $S =< V, C, \tau, P >$, где $V = \{$здоровье $\to 100,$ сила $\to 15,$ опыт $\to 200\}$,
$C = \{$здоровье $\geqslant 0,$ сила $> 0,$ опыт $\geqslant 0\}$,
$\tau = \tau_0$,
$P = \{$вероятность_след_уровня : 0.7$\}$.

Это состояние можно интерпретировать следующим образом: в момент времени t_0 игрок имеет 100 единиц здоровья, 15 единиц силы и 200 единиц опыта. Существуют ограничения на значения атрибутов: здоровье и опыт не могут быть отрицательными, а сила должна быть строго положительной. Вероятность достижения следующего уровня в текущем состоянии составляет 0.75.

Такое представление позволяет точно описать как общие характеристики сущности «игрок», так и ее конкретное состояние в определенный момент времени, включая вероятностные аспекты.

2.2.4. Темпоральные аспекты. Для моделирования изменения состояний во времени введём оператор $Next(S)$, который определяет следующее состояние сущности $Next(S) = S'$, где S' – новое состояние после некоторого игрового события или действия (оператор $Next(S)$ соответствует темпоральному оператору $Next(X)$, описанному в разделе 3.3.1. Таким образом, $Next(S) = S'$ эквивалентно XS в нотации темпоральной логики).

2.2.5. Вероятностные аспекты. Для учета вероятностных элементов введём функцию $P(S \to S')$, которая определяет вероятность перехода из состояния S в состояние S'.

2.2.6. Формальная верификация. Использовав это представление, мы можем сформулировать и проверить свойства игровых сущностей. Например:

$$\forall(\text{здоровье}(t) > 0) \text{ – игрок всегда имеет положительное здоровье,}$$

$$G(\text{опыт}(t+1) \geqslant \text{опыт}(t)) \text{ – опыт игрока не уменьшается со временем.}$$

2.2.7. Пример комплексного определения игровой сущности. Character $= \langle I\colon$ "player_001", $A : \{$health_max, strength, speed$\}$, $S : \langle V : \{$health_max $\to$ 100, strength $\to$ 20, speed $\to$ 5, current_health $\to$ 75, position $\to$ (10, 0, 5)$\}$, $C : \{$health_max > 0, strength > 0, speed > 0, current_health $\leqslant$ health_max$\}$, $\tau\colon$ t_current, $P : \{$poison_active: true$\}\rangle$, $T : \{$health_regen: 1/s, ability_cooldown: $\{$fireball: 30s$\}\}$, $B : \{$crit_chance: 0.1, dodge_probability: 0.05$\}\rangle$

Такое представление позволяет точно описать сущность, ее текущее состояние, включая возможное поведение, временные характеристики и вероятностные аспекты поведения – всё, что является основой для дальнейшего анализа и моделирования игровой системы в рамках FAST-GM.

2.3. Темпоральные операторы для описания эволюции сущностей. Для описания изменения состояний сущностей во времени в модели FAST-GM использован набор темпоральных операторов, основанных на линейной темпоральной логике (LTL) [15]. Эти операторы позволяют формально описывать и анализировать динамику игровых сущностей.

2.3.1. Определение базового набора темпоральных операторов.

- $Next(X)\colon X_\varphi$ означает, что свойство φ будет истинным в следующем состоянии.
- $Always(G)\colon G_\varphi$ означает, что свойство φ будет истинным во всех будущих состояниях.
- $Eventually(F)\colon F_\varphi$ означает, что свойство φ станет истинным в некотором будущем состоянии.
- $Until(U)\colon \varphi U \psi$ означает, что свойство φ будет истинным до тех пор, пока не станет истинным свойство ψ.

2.3.2. Формальное описание изменения состояний сущностей во времени. Пусть $S(t)$ обозначает состояние сущности в момент времени t, тогда можно привести следующие примеры для описания различных переходов от одного времени к другому:

- $XS(t) = S(t+1)\colon$ следующее состояние сущности,
- $G(S(t).health > 0)\colon$ здоровье сущности всегда положительно,
- $F(S(t).level = 10)\colon$ в какой-то момент в будущем сущность достигнет 10 уровня,
- $(S(t).status =$ "poisoned"$)U(S(t).status =$ "normal"$)\colon$ сущность будет отравлена, пока не вернется в нормальное состояние.

2.3.3. Темпоральные операторы для типичных игровых сценариев. Приведём примеры использования темпоральных операторов для типичных игровых сценариев.

- Регенерация здоровья:
  $G(S(t).health < S(t).max\_health \rightarrow XS(t).health > S(t).health)$
  – "всегда, если текущее здоровье меньше максимального, в следующий момент времени оно увеличится".

- Откат способности:
  $G(S(t).ability\_used \rightarrow FS(t).ability\_available)$
  – "всегда после использования способности, в конечном итоге она снова станет доступной".

- Прогресс квеста:
  $(S(t).quest\_status = \text{“}in\_progress\text{”})U(S(t).quest\_status = \text{“}completed\text{”})$
  – "квест будет в процессе выполнения, пока не будет завершен".

- Временный бафф:
  $G(S(t).buff\_applied \rightarrow F(S(t).buff\_duration = 0))$
  – "всегда, если применен бафф, в конечном итоге его длительность станет равной нулю".

- Периодическое событие:
  $G(F(S(t).event\_trigger = true))$
  – "всегда верно, что в конечном итоге произойдет событие (и это будет повторяться)".

Использование этих темпоральных операторов позволяет формально описать сложные временные зависимости и поведение игровых сущностей. Это дает возможность не только смоделировать игровые механики, но и провести формальную верификацию игровых сценариев, обеспечив корректность и предсказуемость поведения игровой системы во времени.

2.4. Вероятностные элементы в модели сущностей. Вероятностные элементы играют ключевую роль в моделировании многих аспектов игровых систем – от случайных событий до неопределенностей в поведении сущностей. В модели FAST-GM мы интегрируем вероятностные элементы, использовав концепции из теории вероятностей и стохастических процессов [16].

2.4.1. Введение вероятностных функций и распределений. Вероятностные элементы (дискретные вероятности / непрерывные распределения / условные вероятности) представлены следующим образом:

- дискретные вероятности: $P(X = x)$, где X – случайная величина, x – конкретное значение.
  Пример: $P(\text{критический\_удар} = true) = 0.1$;

- непрерывные распределения: $f(x)$, где f – функция плотности вероятности.
  Пример: урон может быть распределен нормально с параметрами μ (среднее) и σ (стандартное отклонение);

- условные вероятности: $P(A|B)$, вероятность события A при условии B.
  Пример: $P(\text{победа} \mid \text{уровень\_игрока} > 10) = 0.7$.

2.4.2. Моделирование случайных событий и неопределенностей в поведении сущностей. Приведём примеры, как могут быть представлены случайные события и неопределенности в поведении сущностей.

- Генерация случайных событий:

$$Event(e) = \{true, \text{если } rand() < P(e); false, \text{иначе}\}.$$

Например, генерация случайного события «критический удар»:

$$Critical\_Hit() = \{true, \text{если } rand() < character.crit\_chance; false, \text{иначе}\}.$$

- Стохастические изменения состояния:

$$S(t+1) = f(S(t), \varepsilon), \text{где } \varepsilon - \text{случайная величина.}$$

Например, изменение в следующий тик времени уровня здоровья с учетом заложенного уровня регенерации:

$$character.health(t{+}1) = character.health(t){+}regeneration\_rate{+}Normal(0, 5).$$

- Марковские цепи для моделирования поведения NPC:

$$P(S(t+1) = s_j | S(t) = s_i) = p_{ij}.$$

Например, вероятность изменения поведения NPC в следующий тик времени:

$$P(NPC\_state(t+1) = \text{“атака”} \,|NPC\_state(t) = \text{“патруль”}) = 0.3.$$

2.4.3. Интеграция вероятностных элементов с темпоральными аспектами. Для объединения вероятностных и темпоральных аспектов мы используем концепции из вероятностной темпоральной логики [17]:

- $P_{=r}[\varphi]$: вероятность того, что формула φ истинна, равна r.
  Пример: $P_{=0.8}[F(character.level \geqslant 10)]$ – “вероятность того, что в конечном итоге персонаж достигнет 10 уровня, равна 0.8”;

- $P_{\geqslant r}[\varphi]$: вероятность того, что формула φ истинна, не менее r.
  Пример: $P_{\geqslant 0.95}[G(item.durability > 0)]$ – “с вероятностью не менее 0.95 предмет никогда не сломается”;

- $E[reward]$: ожидаемое значение награды.
  Пример: $E[G(quest.completion\_reward)] \geqslant 100$ – “ожидаемая награда за выполнение квеста всегда не менее 100”.

Использование этих конструкций позволяет формально описать и проанализировать сложные игровые механики, включающие как временные, так и вероятностные аспекты.

Пример комплексного описания игровой механики:

$$P_{\geqslant 0.7}[G(character.health > 0)U(boss.health = 0)]$$

– “с вероятностью не менее 0.7 персонаж останется жив до победы над боссом”.

Интеграция вероятностных элементов в модель FAST-GM позволяет более точно описывать реалистичное поведение игровых сущностей, учитывая элементы случайности и неопределенности, присущие многим игровым системам.

2.5. Взаимодействие между сущностями. Взаимодействие между игровыми сущностями является ключевым аспектом любой игровой системы. В модели FAST-GM мы формализуем различные типы взаимодействий, учитывая их временные и вероятностные характеристики.

2.5.1. Формальное определение типов взаимодействий. В нашей модели мы выделяем следующие типы взаимодействий:

- прямые взаимодействия: $I\_d(E_1, E_2)$ – непосредственное влияние сущности E_1 на E_2;
- косвенные взаимодействия: $I\_i(E_1, E_2, E_3)$ – влияние E_1 на E_2 через посредника E_3;
- синхронные взаимодействия: $I\_s(E_1, E_2, t)$ – взаимодействие, происходящее в конкретный момент времени t;
- асинхронные взаимодействия: $I\_a(E_1, E_2, \Delta t)$ – взаимодействие, растянутое во времени на интервал Δt.

2.5.2. Моделирование влияния взаимодействий на состояния сущностей. Влияние взаимодействия на состояние сущности можно описать как функцию перехода $S'(E) = f(S(E), I(E_1, E_2), t)$, где $S(E)$ – текущее состояние сущности E, $S'(E)$ – новое состояние после взаимодействия, $I(E_1, E_2)$ – взаимодействие между сущностями E_1 и E_2, а t – момент времени.

$$Player.health' = f(Player.health, I_d(Enemy, Player), t) = \\ = \max(0, Player.health - Enemy.damage * (1 - Player.defense/100))$$

2.5.3. Описание сложных взаимодействий с использованием темпоральных и вероятностных элементов. Для описания сложных взаимодействий мы комбинируем темпоральные операторы и вероятностные элементы:

- отложенное воздействие:
  $X[\Delta t]I\_d(Potion, Player.health)$ – "через Δt единиц времени зелье повлияет на здоровье игрока";
- вероятностное взаимодействие:
  $P_{0.3}[I\_d(Player.attack, Enemy.health)]$ – "атака игрока с вероятностью 0.3 повлияет на здоровье врага";
- периодическое взаимодействие:
  $G[F[I\_d(Environment.radiation, Player.health)]]$ – "всегда в будущем будет момент, когда радиация окружения повлияет на здоровье игрока";
- условное взаимодействие:
  $(Player.level \geqslant 10) \rightarrow I\_d(Player, Environment.special\_zone)$ – "если уровень игрока не менее 10, то он может взаимодействовать со специальной зоной".

2.5.4. Пример комплексного взаимодействия. Рассмотрим сценарий «Отравленный клинок» в ролевой игре:

$$I\_poison\_blade = \{trigger : I\_d(Player.weapon, Enemy), effect : P_{0.2}[G[0,30](X Enemy.health' = Enemy.health - 5)], duration : 30 seconds\}$$

Листинг 2

Пример использования навыка «Отравленный клинок»

```
I_poison_blade = {
    trigger: I_d(Player.weapon, Enemy),
    effect: P_0.2[G[0,30] (X Enemy.health'
    = Enemy.health - 5)],
    duration: 30 seconds
}
```

Это взаимодействие означает:

- триггер: прямое взаимодействие оружия игрока с врагом,
- эффект: с вероятностью 0.2 в течение следующих 30 секунд каждую секунду враг будет терять 5 единиц здоровья,
- длительность эффекта: 30 секунд.

Такое формальное описание позволяет точно моделировать сложные игровые механики, учитывая их временные, вероятностные и условные аспекты. Это обеспечивает основу для анализа баланса, выявления потенциальных проблем и оптимизации игрового процесса. Интеграция взаимодействий в модель FAST-GM позволяет создавать детальные и точные представления игровых систем, что особенно важно для сложных многопользовательских игр и игр с открытым миром, где взаимодействия между сущностями могут быть чрезвычайно разнообразными и комплексными.

2.6. Масштабируемость и адаптивность модели. Модель FAST-GM разработана с учетом необходимости ее применения к различным масштабам игровых систем и адаптации к специфическим требованиям различных игровых жанров [18]. В этом разделе мы рассмотрим, как модель обеспечивает масштабируемость и адаптивность.

2.6.1. Применимость модели к различным масштабам игровых систем. FAST-GM может быть применена к игровым системам различного масштаба:

A. Микроуровень. Моделирование отдельных игровых механик или взаимодействий.
   Пример: детальное описание системы крафтинга в игре.

B. Мезоуровень. Описание подсистем или групп взаимодействующих сущностей.
   Пример: моделирование экономики виртуального города.

C. Макроуровень. Моделирование целых игровых миров или глобальных игровых систем.
   Пример: описание эволюции фракций в масштабной стратегической игре.

Для обеспечения масштабируемости используется иерархический подход:

$$E\_macro = \{E\_meso_1, E\_meso_2, ..., E\_meso_n\} E\_meso = \\ = \{E\_micro_1, E\_micro_2, ..., E\_micro_m\}),$$

где E_macro – макросущность, E_meso – мезосущности, E_micro – микросущности.

2.6.2. Адаптация модели к различным игровым жанрам. Адаптивность FAST-GM к различным игровым жанрам [18] является ключевым фактором её универсальности и практической применимости. Для обеспечения масштабируемости и гибкости модели мы предлагаем следующие стратегии и механизмы.

A. Оптимизация обработки событий:

a) иерархическое представление событий – группировка микрособытий в макрособытия для снижения вычислительной нагрузки;

b) приоритизация событий – обработка наиболее критичных событий в реальном времени, остальные – в фоновом режиме;

c) использование приближенных методов – для жанров, где точность менее критична (например, казуальные игры), применение упрощенных моделей.

B. Жанрово-специфическая адаптация:

a) жанровые шаблоны – использование предопределенных наборов сущностей, атрибутов и взаимодействий, характерных для конкретных жанров.

$$RPG\_Template = \{Entities : \{Character, NPC, Item, Quest\}, \\ Attributes : \{level, experience, inventory\}, \\ Interactions : \{combat, dialog, trade\}\}$$

$$RTS\_Template = \{Entities : \{Unit, Building, Resource\}, \\ Attributes : \{health, attack\_power, build\_time\}, \\ Interactions : \{gather, construct, attack\}\}$$

b) расширяемые множества операторов – возможность определения новых темпоральных и вероятностных операторов для специфических игровых механик.

$$Combo\_Operator(moves) = X[t_1]move_1 \wedge X[t_2]move_2 \wedge ... \wedge X[t_n]move_n$$

$$Respawn\_Operator(player) = G(player.health = 0 \rightarrow \\ \rightarrow F[respawn\_time]player.health > 0)$$

C. Параметризация модели – для отражения особенностей конкретного жанра.

$$Simulation\_Precision = time\_step : 0.01, physics\_iterations : 10$$

$$Card\_Game\_Parameters = \\ = deck\_size : 60, max\_hand\_size : 7, turn\_per\_round : 1$$

Эти механизмы адаптации позволяют FAST-GM эффективно моделировать широкий спектр игровых жанров от быстрых аркадных игр до сложных стратегий и ролевых игр с открытым миром. Гибкость модели обеспечивает её применимость как к существующим, так и к новым, формирующимся игровым жанрам, делая FAST-GM мощным инструментом для анализа и разработки разнообразных игровых систем.

2.6.3. Примеры расширения модели для сложных игровых сценариев.

$$\begin{aligned} OpenWorld = \{&World\_State : \{time, weather, global\_events\}, \\ &Dynamic\_Quests : f(Player\_Actions, World\_State) \to Quest, \\ &Emergent\_Behavior : P[G(\forall e \in Entities : Unexpected\_Interaction(e))] > 0\} \end{aligned}$$

$$\begin{aligned} MMORPG = \{&Server\_Shards : S_1, S_2, ..., S_k, \\ &Cross\_Shard\_Interaction : I(E_1 \in S_i, E_2 \in S_j), \\ &Latency\_Model : delay(Action) = Normal(\mu, \sigma)\} \end{aligned}$$

$$\begin{aligned} Procedural\_Generation = \{&Seed : s, \\ &Generation\_Rules : R = r_1, r_2, ..., r_n, Content : C = f(s, R, t), \\ &Consistency\_Constraint : G(\forall c \in C : Consistent(c, World\_Logic))\} \end{aligned}$$

2.6.4. Обеспечение согласованности при масштабировании. Для обеспечения согласованности модели при масштабировании и адаптации используются следующие принципы:

- инвариантность базовых определений – основные концепции (сущности, состояния, взаимодействия) остаются неизменными на всех уровнях масштаба;
- композиционность – сложные системы могут быть построены из более простых компонентов с сохранением свойств и ограничений;
- абстракция и детализация – возможность представления систем на разных уровнях абстракции с сохранением ключевых свойств.

Пример:

$$Abstract\_Combat(E_1, E_2) = P[Win(E_1)] = f(E_1.power, E_2.power)$$

$$\begin{aligned} &\text{Detailed\_Combat}(E_1, E_2) = \\ &\qquad = Sequence\langle Attack(E_1, E_2), Defence(E_2), Counter(E_2, E_1), ...\rangle \end{aligned}$$

Здесь Abstract _ Combat представляет высокоуровневую абстракцию боевого взаимодействия, где результат определяется вероятностной функцией от силы участников, а Detailed _ Combat показывает более детальную последовательность действий в бою.

Масштабируемость и адаптивность FAST-GM позволяют применять модель к широкому спектру игровых проектов, от простых мобильных игр до сложных многопользовательских миров, сохраняя при этом формальную строгость и аналитическую мощь подхода.

2.7. Интеграция компонентов FAST-GM. Модель FAST-GM представляет собой комплексный подход к формальному описанию игровых систем, интегрируя различные аспекты игрового дизайна и механик. Ниже рассмотрим, как основные компоненты модели взаимодействуют между собой.

2.7.1. Сущности и их состояния. Каждая игровая сущность E определяется как $E = \langle I, A, S, T, B \rangle$, где $S = \langle V, C, \tau, P \rangle$. Это позволяет описать как статические характеристики сущностей (через атрибуты A), так и их динамическое поведение (через состояние S).

2.7.2. Темпоральные аспекты. Временная эволюция сущностей описывается с помощью темпоральных операторов (X, G, F, U). Эти операторы связывают текущее состояние сущности с её будущими состояниями, позволяя моделировать изменения во времени.

2.7.3. Вероятностные элементы. Вероятностное поведение сущностей интегрируется через компонент P в определении состояния. Это позволяет моделировать неопределенность и случайность в игровых механиках.

2.7.4. Взаимодействия. Взаимодействия между сущностями описываются функцией $I : E \times E \times T \to S'$, которая учитывает как сами сущности, так и временной аспект их взаимодействия.

2.7.5. Масштабируемость. Иерархическая структура модели (микро-, мезо- и макроуровни) позволяет применять FAST-GM к игровым системам различной сложности и масштаба.

Интеграция этих компонентов позволяет создавать целостные модели игровых систем, учитывающие статические свойства, динамическое поведение, временную эволюцию и вероятностные аспекты игровых сущностей и их взаимодействий. Это обеспечивает мощный инструментарий для анализа, верификации и оптимизации игровых механик на различных этапах разработки игры.

2.8. Формальное описание логик и анализ сложности. В основе FAST-GM лежат линейная темпоральная логика (LTL) и вероятностная логика, которые обеспечивают мощный инструментарий для моделирования динамических аспектов игровых систем. LTL, впервые предложенная Пнуэли [15], позволяет формально описывать и анализировать поведение систем во времени, используя операторы X (Next), G (Globally), F (Future) и U (Until). Вероятностная логика, развитая в работах [19], [20], предоставляет формальный аппарат для работы с неопределенностью и случайностью в игровых механиках. Детальное формальное описание этих логик, включая аксиомы, правила вывода и доказательства ключевых теорем, выходит за рамки данной статьи и будет представлено в отдельной работе.

Интеграция темпоральной и вероятностной логик в FAST-GM существенно повышает выразительность модели, но также увеличивает её вычислительную сложность. Как показано в работе [21], проверка модели для LTL является PSPACE[2]-полной задачей, что накладывает определенные ограничения на применение модели к крупномасштабным игровым системам. Анализ сложности вероятностных вы-

[2]PSPACE-полная задача (или задача PSPACE-полноты) является одной из наиболее трудных задач в классе сложности PSPACE, включающем все задачи, которые могут быть решены с использованием полиномиального объема памяти (или Polynomial Space).

числений, основанный на результатах Пападимитриу [22], указывает на необходимость применения приближенных методов для практической реализации модели.

Для практического применения FAST-GM будут предложены очевидные оптимизации такие, как применение техник символьной верификации [23], ограничение глубины темпорального анализа [24] и использование приближенных методов для вероятностных вычислений [25].

Детальное исследование вычислительной сложности FAST-GM и разработка эффективных алгоритмов для её практического применения являются предметом наших текущих исследований и будут представлены в последующих публикациях.

3. Моделирование взаимодействий между сущностями

3.1. Формальное определение взаимодействий. Взаимодействие между сущностями в FAST-GM определяется как функция (19):

$$I : E \times E \times T \to S',$$

где E – множество сущностей, T – временной домен, S' – новое состояние сущности после взаимодействия.

Формально взаимодействие $I(e_1, e_2, t)$ описывает, как сущность e_1 влияет на сущность e_2 в момент времени t, изменяя ее состояние.

3.2. Графовое представление взаимодействий. Взаимодействия между сущностями можно представить в виде направленного графа (20):

$$G = (V, E),$$

где V – множество вершин, представляющих сущности, а E – множество ребер, представляющих взаимодействия. Каждое ребро $e \in E$ имеет метку (21):

$$l(e) = (I, t, p),$$

где I – тип взаимодействия ($Interaction$), t – временная характеристика, p – вероятность (для стохастических взаимодействий).

3.3. Темпоральные аспекты взаимодействий. Для описания временных аспектов (немедленное / отложенное / периодическое) взаимодействий используются следующие темпоральные операторы:

- $I\_immediate(e_1, e_2)$: немедленное взаимодействие,
- $I\_delayed(e_1, e_2, \Delta t)$: отложенное взаимодействие с задержкой Δt,
- $I\_periodic(e_1, e_2, T)$: периодическое взаимодействие с периодом T.

$$I\_periodic(Sun, Plants, 24h) = \\ = effect : Plants.energy+ = f(Sun.intensity), period : 24h$$

3.4. Вероятностные характеристики взаимодействий. Вероятностные взаимодействия описываются с использованием функции вероятности $P(I|e_1, e_2, t)$ – вероятность взаимодействия I между e_1 и e_2 в момент t.

$$P(Critical\_Hit|Player, Enemy, t) = \\ = Player.critical\_chance + f(Player.luck, Enemy.vulnerability)$$

Заключение

Представлен унифицированный формальный подход к моделированию игровых сущностей и их взаимодействий – FAST-GM. Этот подход объединяет темпоральные и вероятностные аспекты игровых систем, обеспечивая комплексное описание динамики игровых механик. Основные достижения проведенного исследования включают разработку формальной модели FAST-GM, способной описывать широкий спектр игровых сущностей и их взаимодействий, метод интеграции темпоральных и вероятностных элементов в единую модель, а также демонстрацию универсальности и масштабируемости модели для различных игровых жанров и систем разной сложности.

По сравнению с существующими подходами, FAST-GM отличается высокой степенью формальной строгости и интеграцией темпоральных и вероятностных аспектов в единую модель. Это открывает новые возможности для автоматизированного анализа и верификации игровых механик, включая количественную оценку игрового баланса, формальную верификацию игровых сценариев и автоматизированную генерацию тестовых случаев. Такой подход позволяет более эффективно выявлять потенциальные дисбалансы, ошибки и несоответствия в игровой логике, что особенно важно в контексте растущей сложности современных игровых систем.

Практическая реализация FAST-GM сопряжена с рядом технических вызовов, требующих инновационных решений. Для управления большими объемами данных предлагается использование распределенных систем хранения и применение техник анализа больших данных. Оптимизация вычислений может быть достигнута путем параллельной обработки и использования GPU для ускорения вероятностных вычислений. Интеграция FAST-GM с существующими игровыми движками потребует разработки специализированных интерфейсов и адаптеров. Решение этих задач откроет путь к широкому применению FAST-GM в реальных проектах разработки игр.

Несмотря на свои преимущества, модель FAST-GM имеет ряд ограничений, в частности, высокую вычислительную сложность для больших игровых миров и потенциальные трудности в реализации для некоторых игровых жанров. Будущие исследования будут направлены на разработку более эффективных алгоритмов для темпорального и вероятностного анализа, создание специализированных инструментов для работы с FAST-GM, а также расширение модели для более полного охвата эмерджентных явлений в играх. Планируется провести обширное эмпирическое тестирование модели на реальных игровых проектах разных масштабов и жанров, а также исследовать возможности применения машинного обучения для автоматизации анализа и оптимизации игровых систем на основе данной модели.

Практическое применение модели может быть дополнительно усилено использованием структурированного корпуса текстов видеоигр [3], что позволит более эффективно извлекать и анализировать игровые элементы и механики.

В заключение отметим, что предложенный подход представляет собой мощный инструмент для анализа, верификации и оптимизации игровых механик, способный значительно повысить качество и эффективность разработки игр, открывая новые горизонты в game studies и индустрии разработки игр.

Благодарности. Работа выполнена за счет средств Программы стратегического академического лидерства Казанского (Приволжского) федерального университета («ПРИОРИТЕТ-2030»).

Конфликт интересов. Авторы заявляют об отсутствии конфликта интересов.

Литература

1. *Сахибгареева Г.Ф., Кугуракова В.В., Большаков Э.С.* Генерация и балансирование игровых механик видеоигр // Науч. сервис в сети Интернет: тр. XXIV Всерос. науч. конф. (19–22 сент. 2022 г., онлайн). М.: ИПМ им. М.В. Келдыша, 2022. С. 455–485. https://doi.org/10.20948/abrau-2022-6.
2. *Сахибгареева Г.Ф., Кугуракова В.В.* Практики балансирования компьютерных игр // Прогр. сист.: теор. и прил. 2022. Т. 13, № 3. С. 255–273. https://doi.org/10.25209/2079-3316-2022-13-3-255-273.
3. *Нурлыгаянов Н.Р., Кугуракова В.В.* Подход к созданию корпуса текстов видеоигр на основе универсальной структуры // Электр. библ. 2024. Т. 27, № 4. С. 578–597. https://doi.org/10.26907/1562-5419-2024-27-4-578-597.
4. *Левенчук А.* Системное мышление. В 2-х т. Т. 1. Екатеринбург: Ridero, 2024. 412 с.
5. *Aarseth E.* Computer game studies, year one // Game Stud. 2001. V. 1, No 1.
6. *Juul J.* Half-Real: Video Games between Real Rules and Fictional Worlds. Cambridge, MA: MIT Press. 2005. 248 p.
7. *Mateas M., Stern A.* Structuring content in the *façade* interactive drama architecture // Proc. Artif. Intell. Interact. Digital Entertainment Conf. 2005. V. 1, No 1. P. 93–98. https://doi.org/10.1609/aiide.v1i1.18722.
8. *Nelson M.J., Mateas M.* Towards a formal game grammar // Proc. 6th Int. Conf. on Cognitive Modeling. 2008. P. 137–143.
9. *Zagal J.P., Mateas M., Fernández-Vara C., Hochhalter B., Lichti N.* Towards an ontological language for game analysis // Proc. DiGRA 2005 Conf.: Changing Views – Worlds in Play. Tampere: DiGRA, 2005. P. 1–13.
10. *Cook D.* The chemistry of game design // Game Developer. 2007. URL: https://www.gamedeveloper.com/design/the-chemistry-of-game-design.
11. *Björk S., Holopainen J.* Patterns in Game Design. Ser.: Game Development Series. Hingham, MA: Charles River Media, 2005. xvii, 423 p.
12. *Brogi A., Canciani A., Soldani J., Wang P.* A Petri net-nased approach to model and analyze the management of cloud applications // Transactions on Petri Nets and Other Models of Concurrency XI / Koutny M., Desel J., Kleijn J. (Eds.). Ser.: Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2016, pp. 28–48. https://doi.org/10.1007/978-3-662-53401-4 _ 2.
13. *Grünvogel S.M.* Formal models and game design // Game Stud. 2005. V. 5, No 1.
14. *Rakhmankulova V., Kugurakova V.* Developing an online tool for balancing video games // Proc. Int. Conf. on Simplicity and Complexity in SMART Automatics and Energy Systems (SMART-SYSTEMS). 2024. P. 1–6.
15. *Pnueli A.* The temporal logic of programs // Proc. 18th Annu. Symp. on Foundations of Computer Science (sfcs 1977). Providence, RI: IEEE, 1977. P. 46–57. https://doi.org/10.1109/SFCS.1977.32.
16. *Ross S.M.* Introduction to Probability Models. 12th ed. Acad. Press, 2019. 842 p. https://doi.org/10.1016/C2017-0-01324-1.
17. *Hansson H., Jonsson B.* A logic for reasoning about time and reliability // Formal Aspects Comput. 1994. V. 6, No 5. P. 512–535. https://doi.org/10.1007/BF01211866.
18. *Apperley T.H.* Genre and game studies: Toward a critical approach to video game genres // Simul. Gaming. 2006. V. 37, No 1. P. 6–23. https://doi.org/10.1177/1046878105282278.
19. *Nilsson N.J.* Probabilistic logic // Artif. Intell. 1986. V. 28, No 1. P. 71–87. https://doi.org/10.1016/0004-3702(86)90031-7.

20. *Fagin R., Halpern J.Y., Megiddo N.* A logic for reasoning about probabilities // Inf. Comput. 1990. V. 87, Nos 1–2. P. 78–128. https://doi.org/10.1016/0890-5401(90)90060-U.

21. *Sistla A.P., Clarke E.M.* The complexity of propositional linear temporal logics // J. ACM. 1985. V. 32, No 3. P. 733–749. https://doi.org/10.1145/3828.3837.

22. *Papadimitriou C.H.* Games against nature // J. Comput. Syst. Sci. 1985. V. 31, No 2. P. 288–301. https://doi.org/10.1016/0022-0000(85)90045-5.

23. *Biere A., Cimatti A., Clarke E., Zhu Y.* Symbolic model checking without BDDs // Tools and Algorithms for the Construction and Analysis of Systems (TACAS'99): Proc. 5th Int. Conf. / Cleaveland W.R. (Ed.). Ser.: Lecture Notes in Computer Science. V. 1579. Berlin, Heidelberg: Springer, 1999. P. 193–207. https://doi.org/10.1007/3-540-49059-0_14.

24. *Biere A., Heljanko K., Junttila T., Latvala T., Schuppan V.* Linear encodings of bounded LTL model checking // Logical Methods Comput. Sci. 2006. V. 2, No 5. P. 1–64. https://doi.org/10.2168/LMCS-2(5:5)2006.

25. *Henriques D., Martins J.G., Zuliani P., Platzer A., Clarke E.M.* Statistical model checking for Markov decision processes // Proc. 2012 9th Int. Conf. on Quantitative Evaluation of Systems. London: IEEE, 2012. P. 84–93. https://doi.org/10.1109/QEST.2012.19.

Поступила в редакцию 31.07.2024
Принята к публикации 5.09.2024

Кугуракова Влада Владимировна, кандидат технических наук, и. о. заведующего кафедрой индустрии разработки видеоигр Института информационных технологий и интеллектуальных систем, руководитель научно-исследовательской лаборатории «Digital Media Lab»

Казанский (Приволжский) федеральный университет
ул. Кремлевская, д. 18, г. Казань, 420008, Россия
E-mail: *vlada.kugurakova@gmail.com*

ISSN 2541–7746 (Print)
ISSN 2500–2198 (Online)

**UCHENYE ZAPISKI KAZANSKOGO UNIVERSITETA.
SERIYA FIZIKO-MATEMATICHESKIE NAUKI
(Proceedings of Kazan University. Physics and Mathematics Series)**

2024, vol. 166, no. 4, pp. 532–554

ORIGINAL ARTICLE
doi: 10.26907/2541-7746.2024.4.532-554

A Formal Approach to Spatio-Temporal Modeling of Game Systems

V.V. Kugurakova

Kazan Federal University, Kazan, 420008 Russia

E-mail: *vlada.kugurakova@gmail.com*

Received July 31, 2024; Accepted September 5, 2024

Abstract

This article introduces FAST-GM (Formal Approach to Spatio-Temporal Game Modeling), a new unified approach to formal modeling of game entities and their interactions that integrates temporal and probabilistic dimensions, thus offering a comprehensive framework for capturing the dynamics of game systems. Built upon the principles of extended temporal logic and probability theory, FAST-GM accurately describes complex game mechanics and how they evolve. A formal definition of game entities was considered. Their states and interactions were explored. The methods for integrating temporal and probabilistic elements into gameplay were discussed. The applicability of FAST-GM for game balancing, formal verification of game scenarios, and automated generation of test cases was analyzed. Its scalability and adaptability in various game genres were assessed. The results obtained show that FAST-GM should advance the formal modeling of game systems, equipping developers with a powerful toolset for analysis, verification, and optimization of game mechanics throughout the process of creating a video game.

Keywords: formal modeling, game entity, game interaction, temporal logic, probabilistic model, game balance analysis, game scenario verification, FAST-GM, game studies, game design

Acknowledgments. This study was supported by the Kazan Federal University Strategic Academic Leadership Program (PRIORITY-2030).

Conflicts of Interest. The authors declare no conflicts of interest.

References

1. Sahibgareeva G.F., Kugurakova V.V., Bolshakov E.S. Video game's mechanics generation and balancing. *Nauch. servis v seti Internet: tr. XXIV Vseros. nauch. konf. (19–22 sent. 2022 g., onlain)* [Scientific Service & Internet: Proc. XXIV All-Russ. Sci. Conf. (September 19–22, 2022, Online)]. Moscow, IPM im. M.V. Keldysha, 2022, pp. 455–485. https://doi.org/10.20948/abrau-2022-6. (In Russian)

2. Sahibgareeva G.F., Kugurakova V.V. Game balance practices. *Program. Sist.: Teor. Prilozh.*, 2022, vol. 13, no. 3, pp. 255–273. https://doi.org/10.25209/2079-3316-2022-13-3-255-273. (In Russian)

3. Nurlygaianov N.R., Kugurakova V.V. A new approach to creating a corpus of video game texts. *Elektron. Bibl.*, 2024, vol. 27, no. 4, pp. 587–597. https://doi.org/10.26907/1562-5419-2024-27-4-578-597. (In Russian)

4. Levenchuk A. *Sistemnoe myshlenie* [Systems Thinking]. Vol. 1. Yekaterinburg, Ridero, 2024. 412 p. (In Russian)

5. Aarseth E. Computer game studies, year one. *Game Stud.*, 2001, vol. 1, no. 1.

6. Juul J. *Half-Real: Video Games between Real Rules and Fictional Worlds*. Cambridge, MA, MIT Press, 2005. 248 p.

7. Mateas M., Stern A. Structuring content in the *façade* interactive drama architecture. *Proc. Artif. Intell. Interact. Digital Entertainment Conf.*, 2005, vol. 1, no. 1, pp. 93–98. https://doi.org/10.1609/aiide.v1i1.18722.

8. Nelson M.J., Mateas M. Towards a formal game grammar. *Proc. 6th Int. Conf. on Cognitive Modeling*, 2008, pp. 137–143.

9. Zagal J.P., Mateas M., Fernández-Vara C., Hochhalter B., Lichti N. Towards an ontological language for game analysis. *Proc. DiGRA 2005 Conf.: Changing Views – Worlds in Play*. Tampere, DiGRA, 2005, pp. 1–13.

10. Cook D. The chemistry of game design. *Game Developer*. 2007. URL: https://www.gamedeveloper.com/design/the-chemistry-of-game-design.

11. Björk S., Holopainen J. *Patterns in Game Design*. Ser.: Game Development Series. Hingham, MA, Charles River Media, 2005. xvii, 423 p.

12. Brogi A., Canciani A., Soldani J., Wang P. A Petri net-nased approach to model and analyze the management of cloud applications. In: Koutny M., Desel J., Kleijn J. (Eds.) *Transactions on Petri Nets and Other Models of Concurrency XI*. Ser.: Lecture Notes in Computer Science. Berlin, Heidelberg, Springer, 2016, pp. 28–48. https://doi.org/10.1007/978-3-662-53401-4_2.

13. Grünvogel S.M. Formal models and game design. *Game Stud.*, 2005, vol. 5, no. 1.

14. Rakhmankulova V., Kugurakova V. Developing an online tool for balancing video games. *Proc. Int. Conf. on Simplicity and Complexity in SMART Automatics and Energy Systems (SMART-SYSTEMS)*, 2024, pp. 1–6.

15. Pnueli A. The temporal logic of programs. *Proc. 18th Annu. Symp. on Foundations of Computer Science (sfcs 1977)*. Providence, RI, IEEE, 1977, pp. 46–57. https://doi.org/10.1109/SFCS.1977.32.

16. Ross S.M. *Introduction to Probability Models*. 12th ed. Acad. Press, 2019. 842 p. https://doi.org/10.1016/C2017-0-01324-1.

17. Hansson H., Jonsson B. A logic for reasoning about time and reliability. *Formal Aspects Comput.*, 1994, vol. 6, no. 5, pp. 512–535. https://doi.org/10.1007/BF01211866.

18. Apperley T.H. Genre and game studies: Toward a critical approach to video game genres. *Simul. Gaming*, 2006, vol. 37, no. 1, pp. 6–23. https://doi.org/10.1177/1046878105282278.

19. Nilsson N.J. Probabilistic logic. *Artif. Intell.*, 1986, vol. 28, no. 1, pp. 71–87. https://doi.org/10.1016/0004-3702(86)90031-7.

20. Fagin R., Halpern J.Y., Megiddo N. A logic for reasoning about probabilities. *Inf. Comput.*, 1990, vol. 87, nos. 1–2, pp. 78–128. https://doi.org/10.1016/0890-5401(90)90060-U.

21. Sistla A.P., Clarke E.M. The complexity of propositional linear temporal logics. *J. ACM*, 1985, vol. 32, no. 3, pp. 733–749. https://doi.org/10.1145/3828.3837.

22. Papadimitriou C.H. Games against nature. *J. Comput. Syst. Sci.*, 1985, vol. 31, no. 2, pp. 288–301. https://doi.org/10.1016/0022-0000(85)90045-5.

23. Biere A., Cimatti A., Clarke E., Zhu Y. Symbolic model checking without BDDs. *Tools and Algorithms for the Construction and Analysis of Systems (TACAS'99): Proc. 5th Int. Conf.* Cleaveland W.R. (Ed.). Ser.: Lecture Notes in Computer Science. Vol. 1579. Berlin, Heidelberg, Springer, 1999, pp. 193–207. https://doi.org/10.1007/3-540-49059-0_14.

24. Biere A., Heljanko K., Junttila T., Latvala T., Schuppan V. Linear encodings of bounded LTL model checking. *Logical Methods Comput. Sci.*, 2006, vol. 2, no. 5, pp. 1–64. https://doi.org/10.2168/LMCS-2(5:5)2006.

25. Henriques D., Martins J.G., Zuliani P., Platzer A., Clarke E.M. Statistical model checking for Markov decision processes. *Proc. 2012 9th Int. Conf. on Quantitative Evaluation of Systems.* London, IEEE, 2012, pp. 84–93. https://doi.org/10.1109/QEST.2012.19.

⟨ ***Для цитирования:*** *Кугуракова В.В.* Формальный подход к пространственно-временному моделированию игровых систем // Учен. зап. Казан. ун-та. Сер. Физ.-матем. науки. 2024. Т. 166, кн. 4. С. 532–554. https://doi.org/10.26907/2541-7746.2024.4.532-554. ⟩

⟨ ***For citation:*** Kugurakova V.V. A formal approach to spatio-temporal modeling of game systems. *Uchenye Zapiski Kazanskogo Universiteta. Seriya Fiziko-Matematicheskie Nauki*, 2024, vol. 166, no. 4, pp. 532–554. https://doi.org/10.26907/2541-7746.2024.4.532-554. (In Russian) ⟩