ФИЗИКО-МАТЕМАТИЧЕСКИЕ НАУКИ

УДК 519.876.5+519.2
DOI 10.34822/1999-7604-2020-4-42-49

СРАВНЕНИЕ БЫСТРОДЕЙСТВИЯ АЛГОРИТМОВ ГЕНЕРАЦИИ ГАММА-РАСПРЕДЕЛЕНИЯ ДЛЯ ИМИТАЦИОННОЙ МОДЕЛИ КОЛОННОГО СТРУЙНО-ЭМУЛЬСИОННОГО РЕАКТОРА

П. А. Сеченов
Сибирский государственный индустриальный университет, Новокузнецк, Россия
E-mail: pavesa89@mail.ru

Изучается быстродействие алгоритмов генерации случайных чисел по гамма-распределению. В имитационной модели колонного струйно-эмульсионного реактора гамма-распределение применяется при создании частиц для таких параметров, как размер и состав веществ в частице. В качестве языка программирования был выбран ActionScript 3.0. Реализовано 7 алгоритмов генерации гамма-распределения: Марсальи и Цанга, Ченга и Фиста (2 версии), Аренса и Дитера, Танизаки, Шмайсера (2 версии) при значении параметра $\alpha > 1$. Для реализации алгоритма Марсальи и Цанга были рассмотрены алгоритмы генерации нормального закона распределения: Бокса – Мюллера, Марсальи – Брея, Девроя; центральная предельная теорема; методы Неймана, Зиккурат. Произведено сравнение генерации гамма-распределения в двух вариантах: без предварительной инициализации начальных значений и с ней. Самым быстрым при $1 < \alpha < 2$ оказался алгоритм Ченга и Фиста. При $\alpha > 2$ – алгоритм Марсальи и Цанга, он устойчив при увеличении параметра гамма-распределения α.

Ключевые слова: гамма-распределение, быстродействие алгоритмов, имитационная модель, струйно-эмульсионный реактор.

SPEED COMPARISON OF GAMMA DISTRIBUTION GENERATION ALGORITHMS FOR SIMULATION MODEL OF COLUMN JET-EMULSION REACTOR

P. A. Sechenov
Siberian State Industrial University, Novokuznetsk, Russia
E-mail: pavesa89@mail.ru

The article studies the performance of algorithms for generating random numbers from the gamma distribution. In the simulation model of the column jet-emulsion reactor, the gamma distribution is used to create particles for such parameters as the size and composition of substances in the particle. ActionScript 3.0 is chosen as the programming language. Seven algorithms for generating the gamma distribution are implemented: Marsaglia and Tsang, Cheng and Fist (two versions), Ahrens and Dieter, Tanizaki, Schmeiser (two versions) with a parameter value $\alpha > 1$. To implement the Marsaglia and Tsang algorithm, algorithms for generating a normal law of distribution are considered: Box-Muller, Marsaglia-Bray, Devroy, central limit theorem, Neumann, the ziggurat method. The comparison of the generation of the gamma distribution in two versions is made: without preliminary initialization of the initial values and with it. The fastest for $1 < \alpha < 2$ was the Cheng and Fist algorithm. For $\alpha > 2$ Marsaglia and Tsang's algorithm is stable with increasing gamma distribution parameter α.

Keywords: gamma distribution, speed of algorithms, simulation model, jet-emulsion reactor.

Введение

Имитационная модель колонного струйно-эмульсионного реактора, отражающая процесс витания частиц шихты и продуктов реакции в вертикальном потоке высокотемператур-

ного несущего газа, предназначена для определения эффективности разделения входных веществ в пространстве реактора при различных конструктивных и режимных параметрах [1]. Имитационная модель включает в себя следующие модели нижнего уровня: плавления частицы, изменения состава шлака, взаимодействия дисперсных частиц, диффузионного перехода на границе шлак – металл. Все модели работают одновременно и отображаются в режиме реального времени. Для отображения большого количества частиц, для добавления новых моделей и механизмов взаимодействия частиц (рис. 1) требуется оптимизация программной реализации и использование быстродействующих алгоритмов.

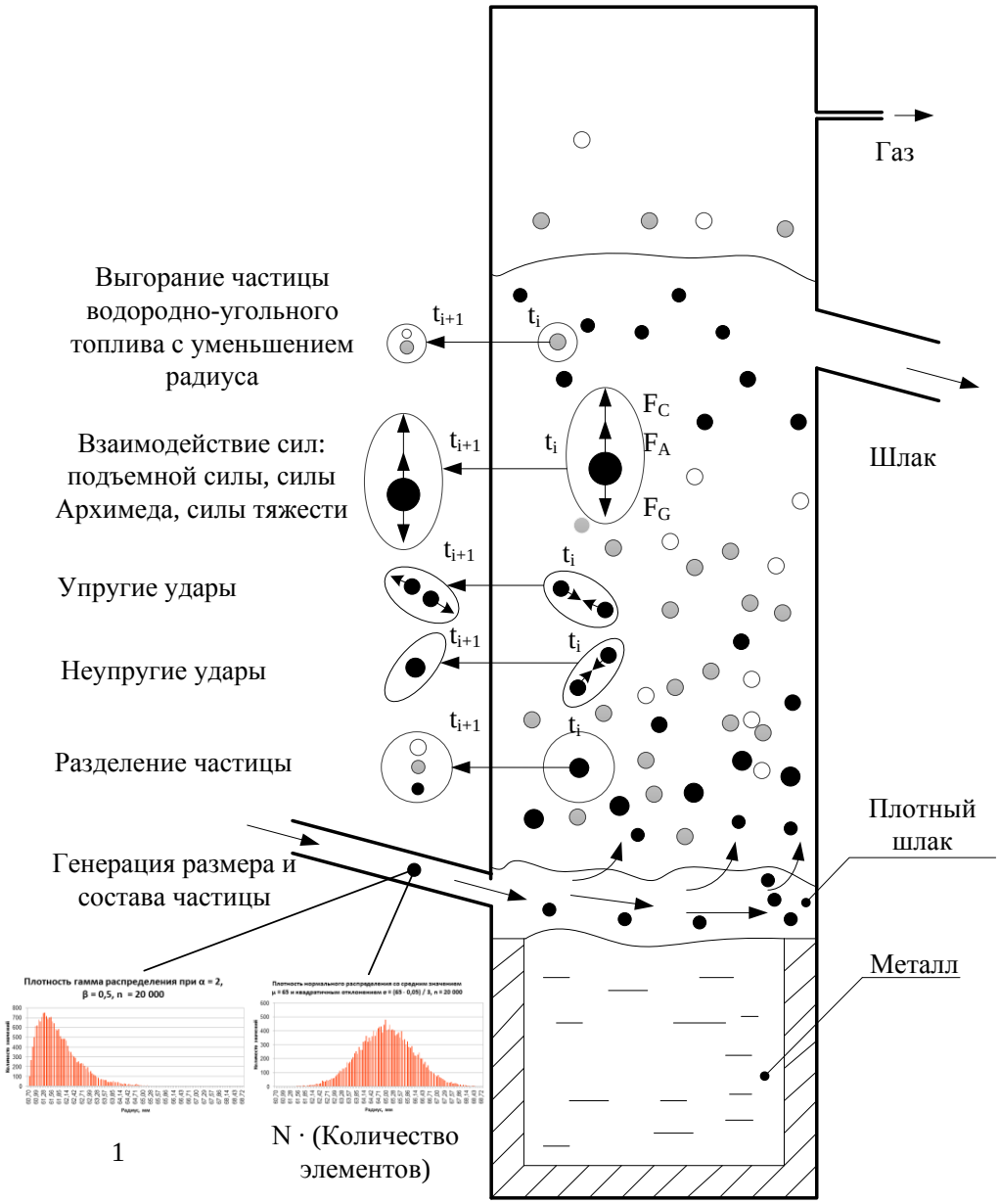

Рис. 1. Взаимодействие между частицами
Примечание: составлено автором.

Как видно из рис. 1, для всех частиц, попадающих в колонный реактор, генерируется гамма-распределение для задания размера радиуса (рис. 2) и нормальное распределение для генерации отклонения состава от средних значений (рис. 3). Например, частица железной руды состоит из 10 веществ: FeO, MnO, SiO_2, CaO, MgO, Al_2O_3, P_2O_5, TiO_2, V_2O_5, Cr_2O_3. Для каждого из этих веществ будет сгенерировано новое значение, отличающееся на ± 5 %

от среднего значения. Например, для оксида железа (при среднем значении 65 %) получим, что значение будет в диапазоне [65 – 65 · 0,05; 65 + 65 · 0,05]. После генерации значений по всем веществам в частице происходит нормализация. В результате получаются частицы разного радиуса, состава и плотности, что, в свою очередь, повлияет на расположение частиц в колонном реакторе.

Рис. 2. Генерация радиуса частиц по гамма-распределению
Примечание: составлено автором.

Рис. 3. Генерация состава частицы по нормальному закону
при среднем значении 65 % и отклонении ± 5 %
Примечание: составлено автором.

Для задания размера и состава частиц используется метод Монте-Карло, реализация которого в виде алгоритма была осуществлена Н. Метрополисом и С. Уламом более 70 лет

назад [2]. Первой работой в этой области считается работа Жоржа-Луи Лекрерка (графа де Бюффона) в 1777 году, в которой были применены метод Монте-Карло и понятие геометрической вероятности для определения числа π [3–4]. Применительно к имитационной модели входными параметрами являются распределения входных частиц по размерам, которые получаются в результате просеивания материалов через стандартный набор сит. Распределение размеров частиц и их состав зависят от происхождения входных продуктов. Для пыли марганцевого производства было получено нормальное распределение [5]. Для железной руды в работе [6] было получено гамма-распределение, поэтому и планируется реализовать быстрый алгоритм генерации случайных величин по гамма-распределению.

Выбор алгоритмов генерации гамма-распределения

В иностранной литературе по изучению метода Монте-Карло [3–4, 7–11] приведены методы генерации по различным законам распределения. Целью данной работы является реализация и сравнение быстродействия алгоритмов для гамма-распределения при $\alpha > 1$. Плотность функции гамма-распределения при различных параметрах α показана на рис. 4.

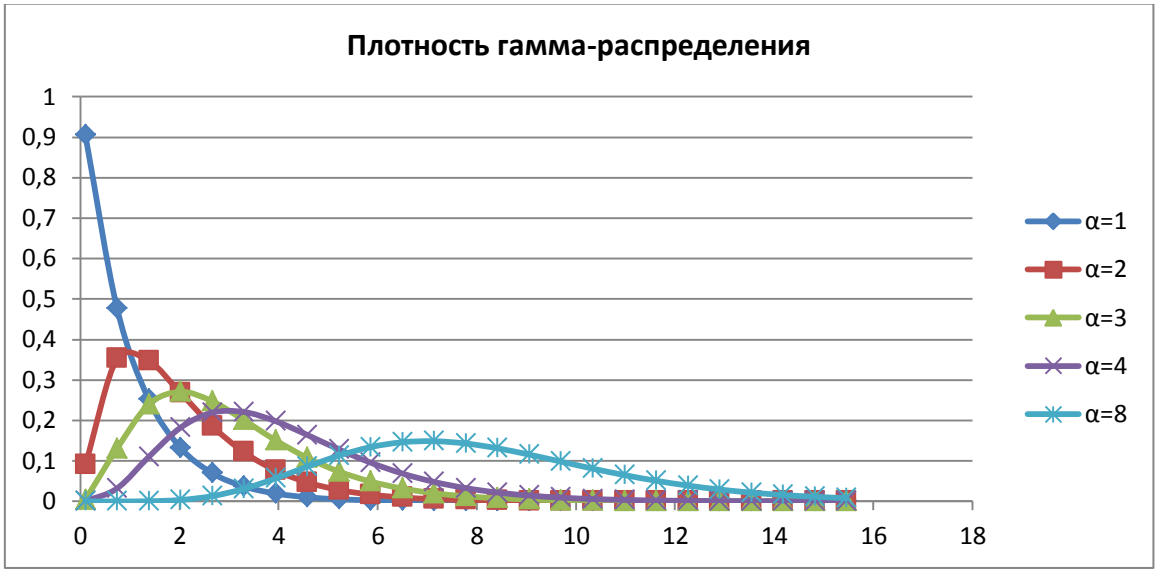

Рис. 4. Плотность гамма распределения при различных параметрах α
Примечание: составлено автором.

В качестве языка программирования был выбран высокоуровневый язык программирования ActionScipt 3.0, который применяется в разнообразных задачах: в системном программировании, в написании web-сценариев, в играх, а также в программировании научных вычислений.

В статье будет рассмотрена реализация пяти алгоритмов генерации гамма-распределения при параметре $\alpha > 1$, а именно:

1) алгоритм Марсальи и Цанга (2000 г.) [4, 8];
2) алгоритм Ченга и Фиста I (1979 г.) [3, 7];
3) алгоритм Аренса и Дитера (1974 г.) [4];
4) алгоритм Ченга и Фиста II (1979 г.) [4];
5) алгоритм Танизаки (2008 г.) [9];
6) алгоритм Шмайсера (1978 г.) G2PE [10];
7) алгоритм Шмайсера (1978 г.) G4PE [10].

Как видно из названия, есть две версии алгоритма Ченга и Фиста далее их будем называть I и II. В численных экспериментах использовались: процессор AMD Phenom II X4 955 Black Edition 3.2 ГГц; операционная система MS Windows 7 Professional SP1 x64. Результаты испытаний быстродействия алгоритмов в среде Adobe Animate CC 2020, написанных на

языке ActionScript 3.0, показаны в табл. 1. Для сравнения быстродействия алгоритмов при различных значениях параметра распределения α генерировался 1 000 000 значений, вызывалась соответствующая функция и замерялось время выполнения в миллисекундах. Первые пять алгоритмов являются легкими. А вот алгоритмы Шмайсера G2PE и G4PE занимают вместе с комментариями 86 и 169 строк соответственно. В основном в статьях рассматривалась реализация алгоритмов на Fortran, C, C++. Алгоритм Шмайсера, где использовался метод сжатия со смешанным распределением, под кодовым названием G4PE оказался самым быстрым в статье самого автора [10] 1978 г. и в статье Танизаки [9] 2008 г. (язык Fortran).

В алгоритме Марсальи и Цанга [4] необходима также генерация случайной величины по нормальному закону распределения. Рассматривались следующие генераторы нормального закона распределения:

1) алгоритм Бокса-Мюллера [4];
2) метод полярных коэффициентов Марсальи и Брея [4];
3) алгоритм генерации неоднородной случайной величины Девроя [4];
4) алгоритм с использованием центральной предельной теоремы [7];
5) метод Неймана;
6) модифицированный метод Зиккурат [11].

Таблица 1

Сравнение быстродействия алгоритмов генерации гамма-распределения

Метод	Время на расчет 10^6 генераций, мс					
	α = 1,01	**α = 2**	**α = 3**	**α = 4**	**α = 8**	**α = 16**
Марсальи и Цанга	1 805	1 787	1 745	1 755	1 712	1 727
Ченга и Фиста I	1 624	1 469	1 596	1 743	2 175	2 911
Аренса и Дитера	3 059	2 303	2 290	2 278	2 305	2 377
Ченга и Фиста II	3 071	2 503	2 797	3 047	2 984	3 004
Танизаки	3 603	3 584	3 536	3 313	3 562	3 371
Шмайсера G2PE	3 934	5 411	5 421	5 368	5 410	3 934
Шмайсера G4PE	4 127	6 876	6 853	6 879	6 780	4 127

Примечание: составлено автором.

В табл. 2 показаны результаты генерации 1 000 000 значений по нормальному закону распределения. Самым быстрым оказался метод Бокса – Мюллера, поэтому и был применен в алгоритме Марсальи и Цанга.

Таблица 2

Сравнение быстродействия алгоритмов генерации нормального распределения

Метод	Время генерации 10^6 значений, мс
Бокса-Мюллера	646
Марсальи-Брея	1 605
Девроя	864
Центральная предельная теорема	4 042
Неймана	2 134
Зиккурат	986

Примечание: составлено автором.

Самым сложным в плане реализации являлся модифицированный алгоритм Зиккурат [11], но он занял всего лишь 3-е место по производительности. Алгоритм Бокса – Мюллера является самым простым (занимает одну строчку) и на языке Action Script 3.0 он оказался самым быстрым.

На графике сравнения быстродействия модельных алгоритмов (рис. 5) видно, что алгоритм Ченга и Фиста I при возрастании значения α начинает заметно замедляться. При $\alpha \leq 4$ самым быстрым алгоритмом является Алгоритм Ченга и Фиста I, при $\alpha > 4$ – алгоритм Марсальи и Цанга.

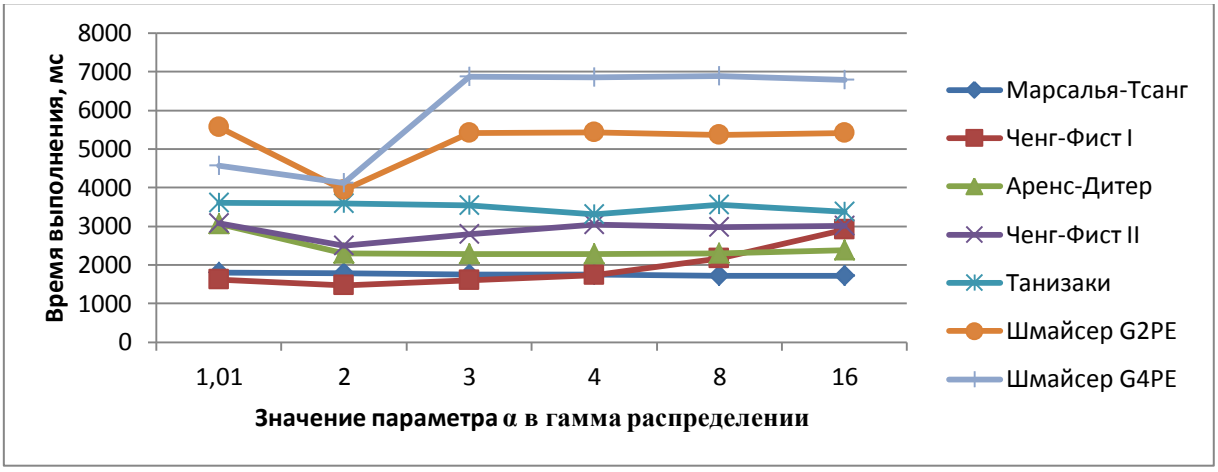

Рис. 5. Сравнительное быстродействие модельных алгоритмов
Примечание: составлено автором.

В статье [9] Х. Танизаки также производит сравнение быстродействия алгоритмов, но инициализация переменных производится один раз, а затем выполняется генерация. Результаты исполнения алгоритмов генерации при одной инициализации начальных значений на языке ActionScript 3.0 представлены в табл. 3, а также на рис. 6.

Таблица 3

Сравнение быстродействия алгоритмов генерации гамма-распределения с предварительной инициализацией

Метод	Время на расчет 10^6 генераций, мс					
	α = 1,01	α = 2	α = 3	α = 4	α = 8	α = 16
Марсальи и Цанга	1 240	1 178	1 199	1 186	1 195	1 218
Ченга и Фиста I	1 216	1 095	1 231	1 329	1 732	2 386
Аренса и Дитера	2 465	1 813	1 782	1 771	1 778	1 819
Ченга и Фиста II	1 762	1 635	2 015	1 987	1 912	1 934
Танизаки	2 129	2 084	2 099	2 092	2 071	2 103
Шмайсера G2PE	1 536	1 965	1 841	1 826	1 797	1 815
Шмайсера G4PE	1 866	1 716	1 512	1 491	1 419	1 418

Примечание: составлено автором.

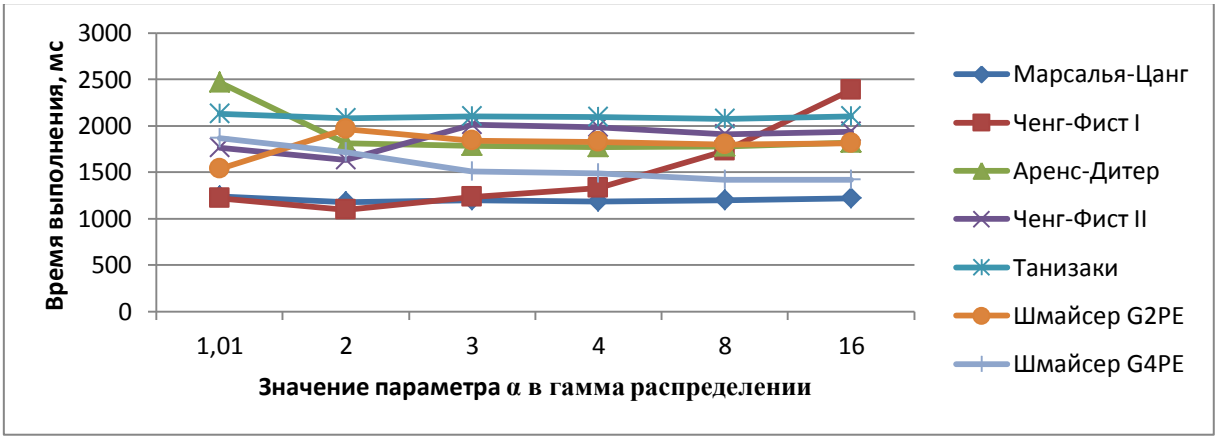

Рис. 6. Сравнительное быстродействие модельных алгоритмов с предварительной инициализацией
Примечание: составлено автором.

С предварительной инициализацией, которая будет выполняться 1 раз (а не 10^6 раз), все алгоритмы ускорились. Самое заметное ускорение произошло у алгоритма Шмайсера

G4PE, и он стал занимать 2–4-е места по быстродействию. Это не удивительно, предварительные вычисления занимают 30 строк. Заметно быстрее стал и алгоритм G2PE. Тенденция к нестабильности у алгоритма Ченга – Фиста I видна и в этом случае: он занимает как первое место при $\alpha < 3$, так и одновременно последнее, например при $\alpha = 16$. Стабильным и самым быстрым является алгоритм Марсальи – Цанга. В этом алгоритме используется генерация случайной величины по нормальному распределению, для чего был применен самый быстрый алгоритм Бокса – Мюллера.

Заключения и выводы

В исследовании реализованы алгоритмы генерации функции гамма-распределения, представленные в книгах [3, 4, 7] и статьях [8–11] на языке ActionScript 3.0. Самым быстрым и устойчивым к изменению оказался алгоритм Марсальи и Цанга. В имитационной модели колонного струйно-эмульсионного реактора планируется реализовать данный алгоритм в виде функции класса расчетов (рис. 7). В результате освободятся вычислительные ресурсы, которые можно задействовать для увеличения количества отображаемых частиц, что повлияет в конечном итоге на точность имитационного моделирования.

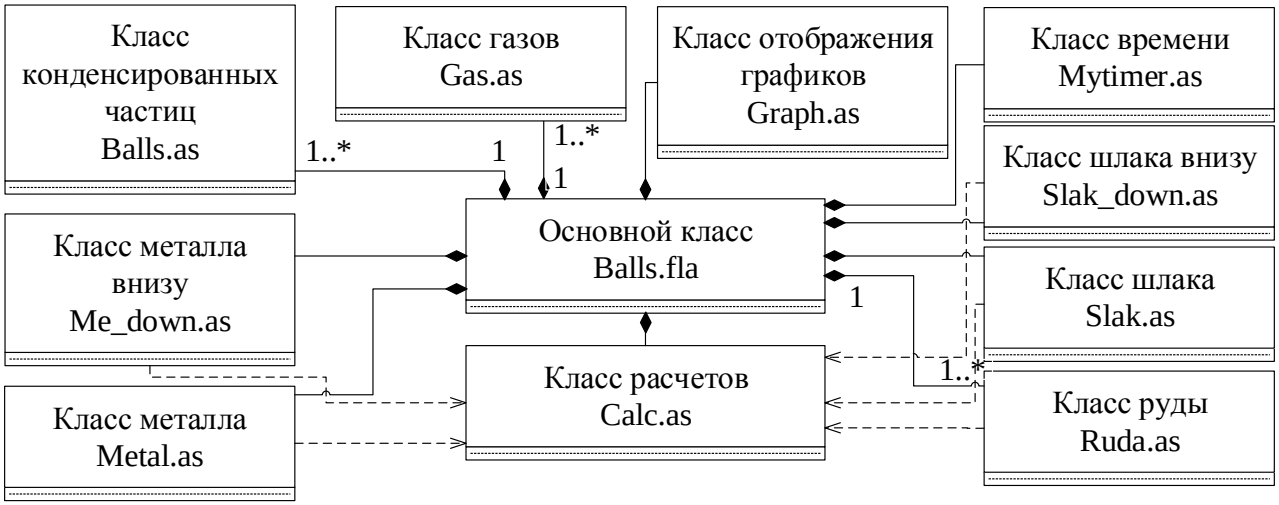

Рис. 7. UML-диаграмма взаимодействия между основным модулем и классами программы
Примечание: составлено автором.

Литература

1. Цымбал В. П., Сеченов П. А., Рыбенко И. А. Имитационное моделирование на основе «первых принципов» и статистическая механика Гиббса // Вестн. Сибир. гос. индустриал. Ун-та. 2020. № 2 (32). С. 54–67.

2. Metropolis N., Ulam S. The Monte Carlo Method // J. Amer. statistical assoc. 1949 Vol. 44, No. 247. P. 335–341.

3. Dunn W. L., Shultis J. K. Exploring Monte Carlo Methods. Elsevier, 2011. 398 p.

4. Kroese D. P., Taimre T., Botev Z. I. Handbook of Monte Carlo Methods. Wiley, 2011. 743 p.

5. Сеченов П. А., Цымбал В. П., Оленников А. А. Имитационная модель разделения составляющих пыли марганцевого производства // Кибернетика и программирование. 2016. № 2. С. 34–41.

6. Ермакова Л. А., Красноперов С. Ю., Калашников С. Н. Математическое моделирование нестационарного тепломассообмена в дисперсных системах // Металлургия: технологии, инновации, качество / под общей ред. Е. В. Протопопова. Новокузнецк, 2015. С. 265–268.

7. Gentle J. E. Random Number Generation and Monte Carlo Methods. New York: Springer. 2nd ed. 2005. 381 p.

8. Marsaglia G., Tsang W. A Simple Method for Generating Gamma Variables // ACM Transactions on Mathematical Software. 2000. No. 26 (3). P. 363–372.

9. Tanizaki H. A Simple Gamma Random Number Generator for Arbitrary Shape Parameters // Economics Bulletin, AccessEcon, 2008. Vol. 3(7). P. 1–10.

10. Schmeiser B. W. Squeeze methods for generate gamma variates. Dalas: Southern Methodist univ. 1978. 21 p.

11. Doornik J. A. An inproved Ziggurat method to generate normal random samples. University of Oxford. 2005. URL: https://www.doornik.com/research/ziggurat.pdf (дата обращения: 08.12.2020).