МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра информационной безопасности

КУРСОВАЯ РАБОТА

по дисциплине «Моделирование систем»

Тема: Анализ подходов к моделированию систем

Студентка гр. 136 _____ Горбунова Д.А.

Преподаватель _____ Шульженко А.Д.

Санкт-Петербург

2024

ЗАДАНИЕ

НА КУРСОВУЮ РАБОТУ

Студентка Горбунова Д.А.

Группа 1361

Тема работы: Анализ подходов к моделированию систем

Исходные данные:

- Количество процессов в системе – 6;
- Количество приоритетов процессов – 6;
- Квант времени, отводимый на выполнение процесса – 136 мс;
- Количество процессоров, выполняющих процессы – 1;

Стохастическая аппроксимация правила обращения процессов к процессору имеет распределение гиперэкспоненциальное.

Содержание пояснительной записки:

«Содержание», «Введение», «Непрерывно-детерминированный подход», «Дискретно-детерминированный подход», «Дискретно-стохастический подход», «Непрерывно-стохастический подход», «Сетевой подход», «Обобщенный подход», «Заключение», «Список использованных источников».

Предполагаемый объем пояснительной записки:

Не менее 00 страниц.

Дата выдачи задания: 16.10.2024

Дата сдачи реферата: 00.00.2000

Дата защиты реферата: 00.00.2000

Студентка _____ Горбунова Д.А.

Преподаватель _____ Шульженко А.Д.

АННОТАЦИЯ

Кратко (в 8-10 строк) указать основное содержание курсового проекта (курсовой работы), методы исследования (разработки), полученные результаты.

SUMMARY

Briefly (8-10 lines) to describe the main content of the course project, research methods, and the results.

СОДЕРЖАНИЕ

	Введение	4
1.	Наименования разделов	5
1.1.		0
1.2.		0
2.		0
2.1.		0
2.2.		0
3.		0
3.1.		0
3.2.		0
	Заключение	0
	Список использованных источников	0
	Приложение А. Название приложения	0

ВВЕДЕНИЕ

Кратко описать цель работы, основные задачи им методы их решения.

1. НЕПРЕРЫВНО-ДЕТЕРМИНИРОВАННЫЙ ПОДХОД

1.1. Краткое теоретическое описание

В данном подходе в качестве математической модели используются дифференциальные уравнения. Обычно в таких математических моделях в качестве независимой переменной, от которой зависят неизвестные искомые функции, служит время t.

$$\vec{y}'(t) = \vec{f}(\vec{y}(t), t),$$
$$\vec{y}(t_0) = \overrightarrow{y_0},$$

где

$$\vec{y}(t) = (y_1, y_2, \dots, y_n), \vec{y}' = \frac{d\vec{y}}{dt},$$

$\vec{f} = (f_1, f_2, \dots, f_n)$ – вектор-функция, непрерывная на некотором (n+1)-мерном множестве.

Так как математические схемы такого вида отражают динамику изучаемой системы, то есть ее поведение во времени, то они называются D-схемами (англ. dynamic). В простейшем случае обыкновенное дифференциальное уравнение имеет вид:

$$y' = f(y, t)$$

1.2. Определение ограничений, накладываемых областью применимости этого подхода

Непрерывно-детерминированный подход позволяет отразить непрерывные процессы в системах и представляет собой метод моделирования и анализа сложных систем, основанный на следующих принципах:

- Непрерывность: переменные системы могут принимать любые значения из заданного диапазона с высокой точностью.
- Детерминированность: результаты вычислений должны быть однозначными для заданных входных данных.
- Физическая обоснованность: модели должны отражать реальные физические процессы и взаимодействия в системе.

- Гибкость: позволяет описывать различные аспекты системы, включая пространственные распределения, временные зависимости и взаимодействия между компонентами.

Ограничения области применения непрерывно-детерминированного подхода включают в себя следующие аспекты:

- Сложность моделирования: лучше подходит для простых систем с небольшим числом переменных. Для сложных систем может потребоваться использование более сложных методов или комбинирование с другими подходами.
- Точность вычислений: при работе с большими числами или высокоточными расчетами может возникнуть проблема с представлением значений в виде дробей. Из-за природы непрерывных чисел могут возникать проблемы с точностью при очень малых значениях или при работе с плавающей точкой.
- Временные затраты: требует значительных вычислительных ресурсов при обработке больших объемов данных. Это ограничивает его применение на устройствах с ограниченной мощностью.
- Ограничение по размеру: менее эффективен для работы с большими массивами данных, так как требует значительного объема памяти.
- Ограничение по скорости: вычисления могут быть медленнее, чем в других методах, особенно для больших систем.
- Ограничения по масштабированию: хорошо масштабируется для увеличения точности, но может столкнуться с трудностями при попытках уменьшить количество переменных.
- Ограничения по интерпретации результатов: иногда сложно интерпретировать результаты, особенно если они не соответствуют ожидаемому поведению системы.

1.3. Доказательство неприменимости подхода к моделированию

Наша система не является непрерывно-детерминированной по нескольким причинам:

1. Количество процессов и приоритетов: В системе имеется 6 процессов и 6 уровней приоритета. Это создает сложную динамику взаимодействия процессов, которая может быть лучше описана с помощью дискретных методов. Непрерывно-детерминированные модели не способны эффективно управлять множеством процессов с различными приоритетами, так как они предполагают более простую физическую структуру взаимодействия

2. Квант времени: Квант времени в 136 миллисекунд указывает на то, что система работает в рамках дискретных временных интервалов. Данный параметр подразумевает, что процессы могут "прыгать" между состояниями, что противоречит основным принципам D-схем, где изменения должны быть непрерывными и плавными

3. Гиперэкспоненциальное распределение: Стохастическая аппроксимация обращения процессов к процессору имеет распределение 4-гиперэкспоненциальное. Это распределение предполагает наличие значительных колебаний в интервалах времени между событиями, что делает систему более подходящей для стохастических или дискретных моделей, где случайные изменения могут быть учтены. Непрерывные модели не могут адекватно учитывать такие вариации

2. ДИСКРЕТНО-ДЕТЕРМИНИРОВАННЫЙ ПОДХОД

2.1. Краткое теоретическое описание

Дискретно-детерминированный подход основан на теории конечных автоматов. Конечный автомат – это автомат с конечным множеством внутренних состояний и входных (а следовательно, и выходных) сигналов.

Дискретно-детерминированный подход используется в тех случаях, когда необходимо смоделировать систему с дискретными состояниями, переходящими из одного состояния в другое в конкретные фиксированные моменты времени, то есть конечный автомат (система) переходит в другое состояние под воздействием внешних сигналов. Каждый переход происходит мгновенно и может рассматриваться как отдельное событие.

Таким образом, дискретно-детерминированный подход позволяет моделировать обработку заявок по приоритетам, а также следить за состоянием процессора и очереди в каждый момент времени.

2.2. Определение ограничений, накладываемых областью применимости этого подхода

Дискретно-детерминированный подход накладывает ряд ограничений на вычислительную систему, а именно:

- Отсутствие случайностей: все переходы между состояниями предсказуемы.
- Мгновенные переходы: переход из одного состояния в другое считается мгновенным.
- Дискретные события: все события происходят в дискретные моменты времени, то есть квант времени фиксирован.

Таким образом, мы может сделать вывод, что дискретно-детерминированный подход применим для нашей абстрактной вычислительной системы, так как явления могут быть рассмотрены как поочередные процессы с определенным шагом.

2.3. Определение цели моделирования

Цель моделирования нашей абстрактной вычислительной системы с использованием дискретно-детерминированного подхода – создание точной и устойчивой математической модели, которая позволяет анализировать и прогнозировать работу системы или процесса с определенными входными данными и правилами поведения.

2.4. Доопределение исходных данных, недостающих для применения этого подхода

Исходные данные с доопределенными данными:

- Количество процессов в системе – 6;
- Количество приоритетов процессов – 6;
- Квант времени, отводимый на выполнение процесса – 136 мс;
- Количество процессоров, выполняющих процессы – 1;
- Стохастическая аппроксимация правила обращения процессов к процессору имеет распределение гиперэкспоненциальное.
- Количество очередей – 1;
- Количество мест в очереди – 8.

2.5. Проведение моделирования

Определим начальное состояние системы S_0:

- Время: t = 0;
- Процессы N[0 … 5] = free;
- Приоритет P_n= free;
- Очередь: BUF[0 … 7] = free.
- Запросы Z_n.

Начальное состояние – это момент, когда система не обрабатывает никаких запросов, то есть очередь и процесс пустые. Начальное состояние системы представлено на рисунке 1.

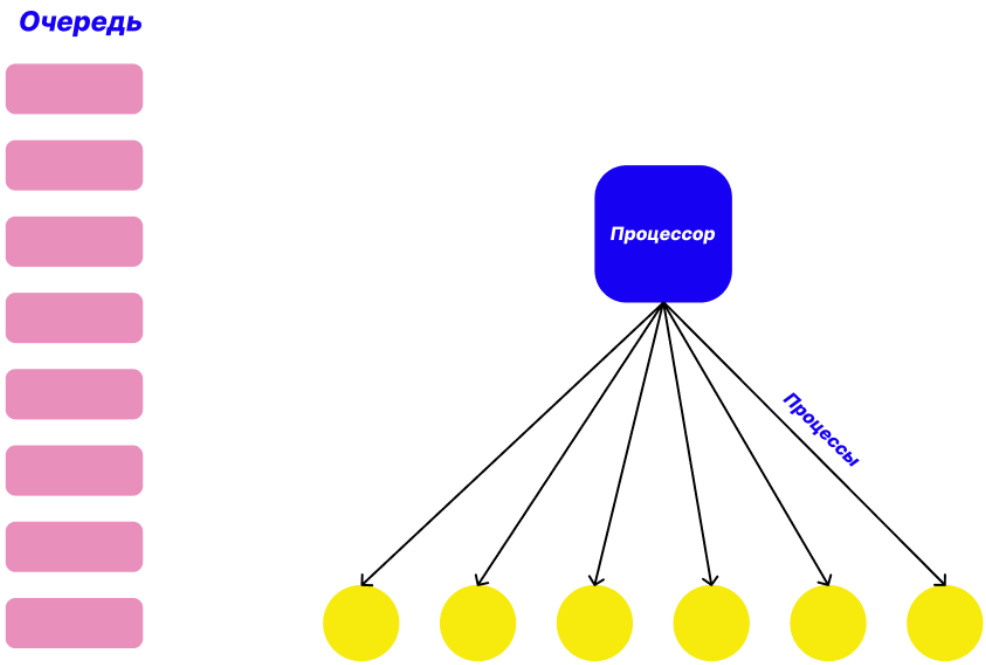

Рисунок 1 – Начальное состояние системы.

Далее в систему поступают запросы. В системе 6 процессов, поэтому одновременно может обрабатываться от 1 до 6 запросов. Время обработки 1 запроса = 136 мс, а время поступления запросов определяется гиперэкспоненциальным распределением и равно Δt_n. Условимся, что запросы не могут поступать одновременно для избежания конфликтных ситуаций, когда одновременно поступают запросы с равными приоритетами, а свободных процессов меньше, чем новых запросов. Также в системе 6 приоритетов, где $p_6 > p_5 > p_4 > p_3 > p_2 > p_1$.

Как только первый запрос поступает, система начинает свою работу.

Рассмотрим поступление первого запроса:

t = 0;

N[0] =$Z_1(p_4)$;

N[1 … 5] = free;

BUF[0 … 7] = free;

$S_0(Z_1) \rightarrow S_1$.

После поступления первого запроса, он занимает процесс, и система переходит в состояние S_1. В таком состоянии процессы частично заняты, а

очередь полностью свободна. Состояние системы S_1 представлено на рисунке 2.

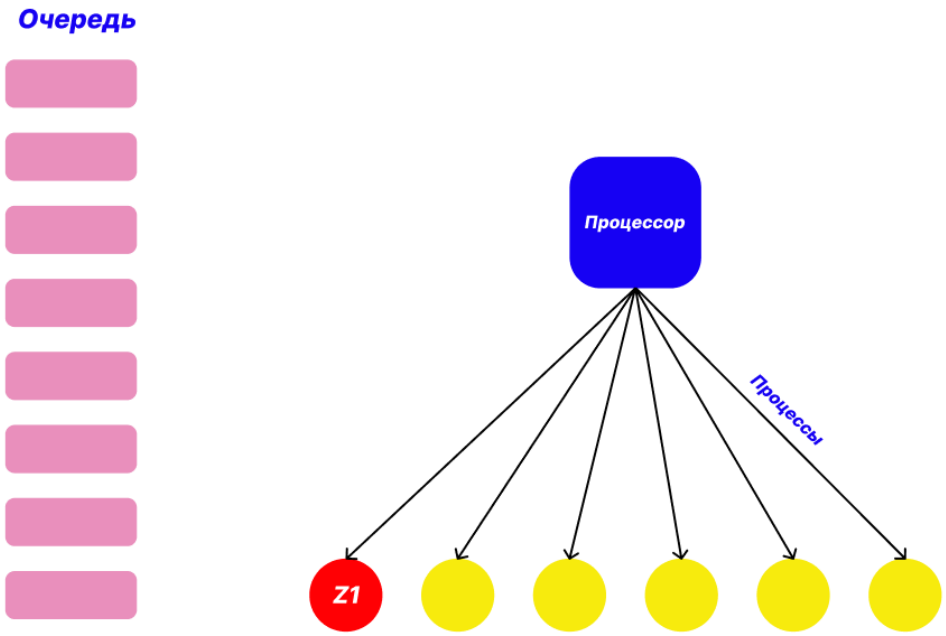

Рисунок 2 – Состояние системы S_1.

Система будет находить в этом состоянии в том случае, если есть свободные потоки. Если еще 5 заявок придет быстрее, чем обработается первый запрос, то система перейдет в состояние S_2, в котором все процессы заняты, а очередь полностью свободна.

Рассмотрим поступление шестого запроса:

t = $\Delta t_1 + \Delta t_2 + \Delta t_3 + \Delta t_4 + \Delta t_5$;

N[0] =$Z_1(p_4)$;

N[1] =$Z_2(p_1)$;

N[2] =$Z_3(p_2)$;

N[3] =$Z_4(p_3)$;

N[4] =$Z_5(p_6)$;

N[5] = $Z_6(p_4)$;

BUF[0 … 7] = free;

$S_1(Z_6) \rightarrow S_2$.

Состояние системы S_2 представлено на рисунке 3.

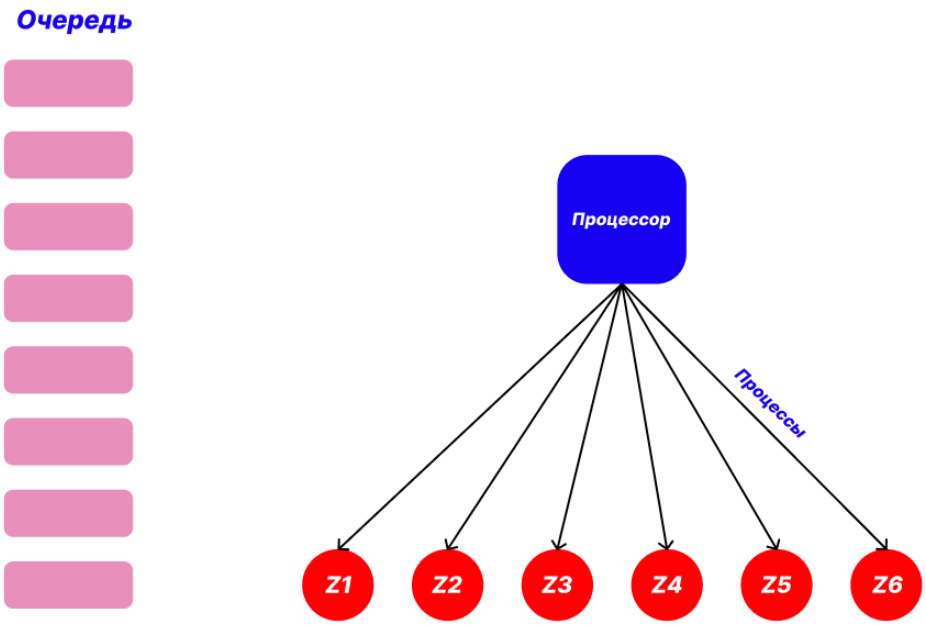

Рисунок 3 – Состояние системы S_2.

Если седьмой запрос поступит в систему быстрее, чем первый запрос выйдет из обработки, он будет помещен в очередь. Рассмотрим поступление 7 запроса:

t = $\Delta t_1 + \Delta t_2 + \Delta t_3 + \Delta t_4 + \Delta t_5 + \Delta t_6$;

N[0] =$Z_1(p_4)$;

N[1] =$Z_2(p_1)$;

N[2] =$Z_3(p_2)$;

N[3] =$Z_4(p_3)$;

N[4] =$Z_5(p_6)$;

N[5] = $Z_6(p_4)$;

BUF[0] = $Z_7(p_2)$;

BUF[1 … 7] = free;

$S_2(Z_7) \rightarrow S_3$.

Система переходит в состояние S_3, в котором потоки полностью заняты, а очередь занята частично. Состояние системы S_3 представлено на рисунке 4.

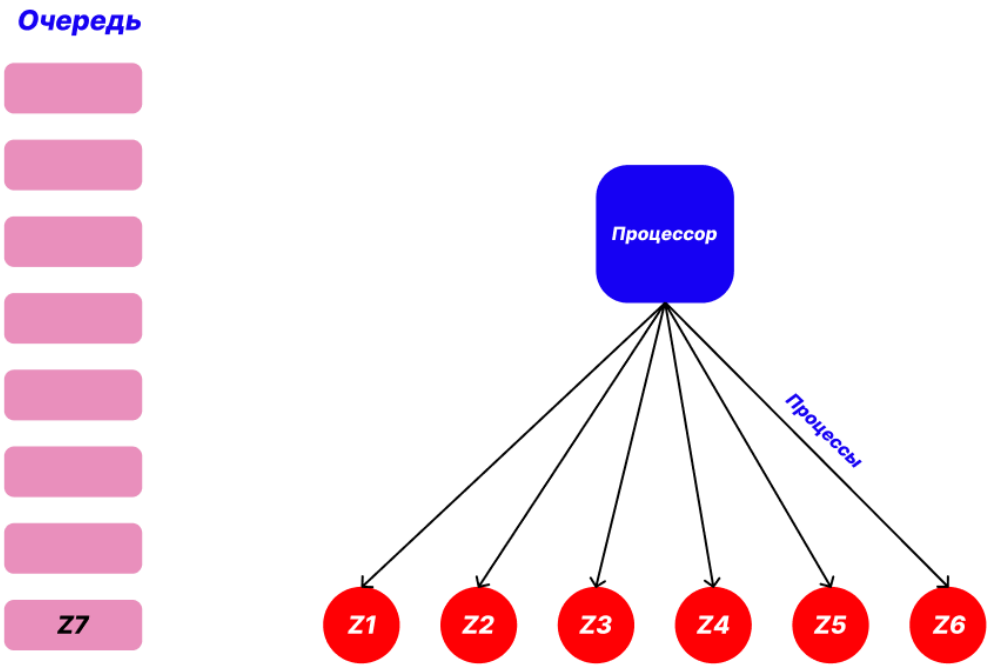

Рисунок 4 – Состояние системы S_3.

Когда все процессы заняты, запросы распределяются по очереди следующим образом: чем выше приоритет запроса, тем ближе к началу очереди он помещается. Как только один из запросов выходит из процесса, тот запрос, который стоит первый в очереди, помещается в обработку, а все запросы, стоящие в очереди, смещаются на одну позицию вперед. Рассмотрим выход из обработки первого запроса:

t = $\Delta t_1 + \Delta t_2 + \Delta t_3 + \Delta t_4 + \Delta t_5 + \Delta t_6 + 136$ мс;

N[0] =$Z_7(p_2)$;

N[1] =$Z_2(p_1)$;

N[2] =$Z_3(p_2)$;

N[3] =$Z_4(p_3)$;

N[4] =$Z_5(p_6)$;

N[5] = $Z_6(p_4)$;

BUF[0 … 7] = free;

$S_3(Z_7) \rightarrow S_2$.

В очереди находился один запрос и после того, как один процесс освободился, запрос из очереди перешел в обработку, а система вернулась в состояние S_2. Текущее состояние системы представлено на рисунке 5.

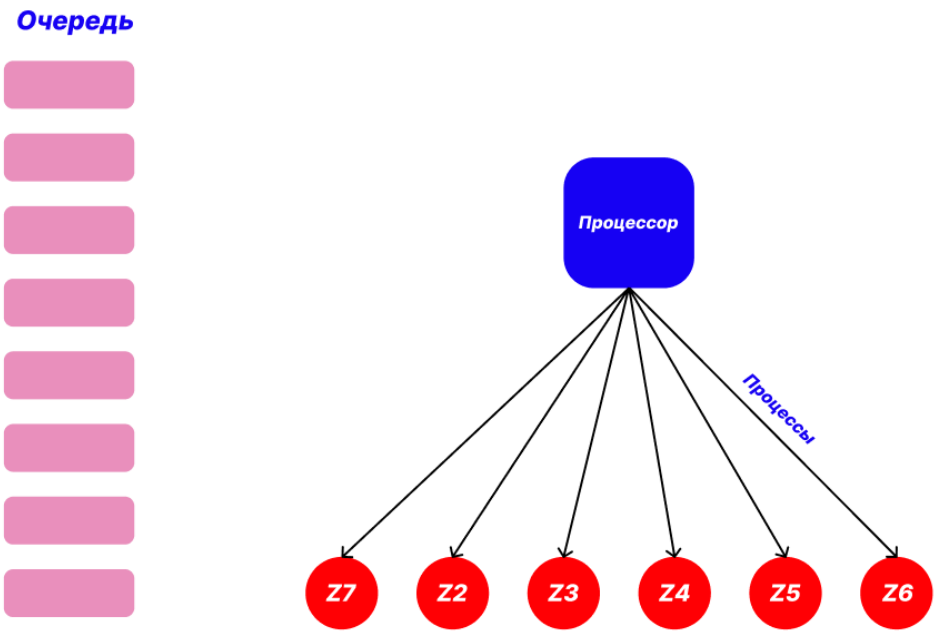

Рисунок 5 – Текущее состояние системы.

Если новые запросы будут поступать в систему, в которой заняты все процессы, они будут помещаться в очередь. Система перейдет в состояние S_3 и будет находиться в этом состоянии, пока в очереди будут оставаться свободные места. Рассмотрим поступление 15 запроса в очередь:

t = $\Delta t_1 + \Delta t_2 + \Delta t_3 + \Delta t_4 + \Delta t_5 + \Delta t_6 + 136 + \Delta t_7 + \Delta t_8 + \Delta t_9 + \Delta t_{10} + \Delta t_{11} + \Delta t_{12} + \Delta t_{13} + \Delta t_{14}$;

N[0] =$Z_7(p_2)$;

N[1] =$Z_2(p_1)$;

N[2] =$Z_3(p_2)$;

N[3] =$Z_4(p_3)$;

N[4] =$Z_5(p_6)$;

N[5] = $Z_6(p_4)$;

BUF[0] = $Z_{15}(p_6)$;

BUF[1] = $Z_8(p_5)$;

BUF[2] = $Z_9(p_5)$;

BUF[3] = $Z_{14}(p_5)$;

BUF[4] = $Z_{10}(p_4)$;

BUF[5] = $Z_{11}(p_3)$;

BUF[6] = $Z_{12}(p_2)$;

BUF[0] = $Z_{13}(p_1)$;

$S_3(Z_{15}) \rightarrow S_4$.

В результате видно, что все запросы помещены в очередь согласно приоритету. Теперь все процессы заняты, все места в очереди заняты и системы перешла в состояние S_4, в котором система полностью занята. Состояние системы S_4 представлено на рисунке 6.

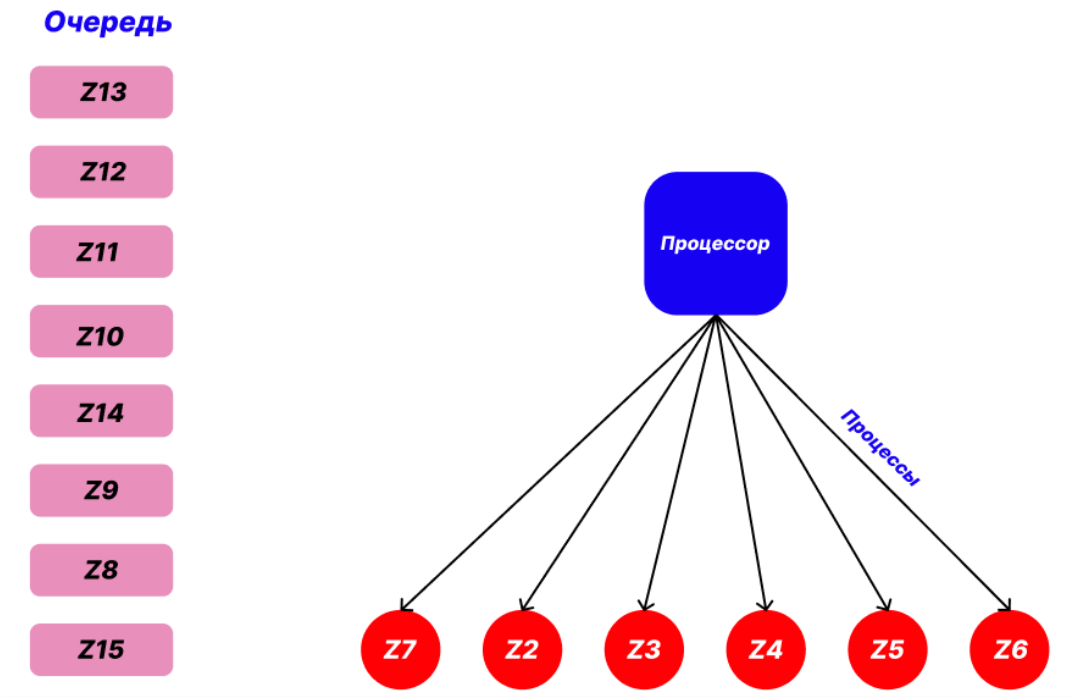

Рисунок 6 – Состояние системы S_4.

Когда очередь полностью занята и в систему поступает новый запрос, то система работает следующим образом: если приоритет нового запроса больше приоритета последнего запроса в очереди, то новый приоритет помещается в очередь согласно своему приоритету, а последний запрос из очереди отклоняется. Если же приоритет нового запроса меньше или равен приоритету последнего запроса в системе, то новый запрос отклоняется.

Система работает следующим образом:

- Полностью пустая система находится в состоянии S_0.
- Когда система полностью пустая и в нее поступает запрос, этот запрос помещается в процесс, а система переходит в состояние S_1.

- Пока в системе в обработке находится от 1 до 5 запросов система сохраняет состояние S_1.
- Если в системе остался один свободный процесс и приходит новый запрос, то этот запрос помещается в процесс и система переходит в состоянии S_2.
- Когда все процессы в системе заняты, а в систему поступает новый запрос, а очередь полностью пустая, то этот запрос помещается в очередь на первую позицию, а система переходит в состояние S_3.
- Пока все процессы заняты, а в очереди есть свободные места, все новые запросы помещаются в очередь согласно своему приоритету и система сохраняется в состоянии S_3.
- Как только последнее место в очереди занято, система переходит в состояние S_4.
- Если система полностью занята и в нее приходит новый запрос, то если его приоритет выше приоритета последнего запроса в очереди, то новый запрос помещается в очередь согласно своему приоритету, а последний запрос из очереди отклоняется. Если же приоритет нового запроса меньше или равен приоритету последнего запроса в очереди, то новый запрос отклоняется.
- Поступление каждого запроса увеличивает время работы системы на Δt_n, которое определяется гиперэкспоненциональным распределением.
- Обработка каждого запроса занимает 136 мс.
- Если очередь полностью занята, а один из запросов выходит из обработки, то первый запрос в очереди переходит в обработку, все запросы в очереди смещаются на одну позицию вперед, а система возвращается в состояние S_3.
- Если в очереди остался один запрос, а один запрос выходит из обработки, то запрос из очереди помещается в процесс, а система возвращается в состояние S_2.

- Если все процессы заняты, а очередь пуста, и один запрос выходит из обработки, то система возвращается в состояние S_1.
- Как только последний запрос выходит из обработки, система возвращается в состояние S_0.

2.6. Анализ результатов

В процессе моделирования были рассмотрены все возможные состояния системы, а именно:

- S_0 – Состояние, в котором система полностью свободна;
- S_1 – Состояние, в котором заняты 1 – 5 процессов, а очередь свободна;
- S_2 – Состояние, в котором процессы заняты, а очередь свободна;
- S_3 – Состояние, в котором процессы заняты, а очередь частично занята.
- S_4 – Состояние, в котором система полностью занята.

Также были рассмотрены все возможные переходы между состояниями, а именно:

- Поступление первого запроса в систему;
- Поступление последующих запросов в систему, когда еще есть свободные процессы;
- Поступление запросов, когда процессы заняты, а в очереди есть места;
- Поступление запросов, когда система полностью занята.

2.7. Выводы

Дискретно-детерминированный подход оказался достаточно эффективным для моделирования моей системы. С помощью этого метода возможно оценить влияние приоритетов и проанализировать поведение системы.

3. ДИСКРЕТНО-СТОХАСТИЧЕСКИЙ ПОДХОД

3.1. Краткое теоретическое описание

Дискретно-стохастический подход – это подход к моделированию систем, в котором переменные принимают случайные значения, то есть данный подход используется в ситуациях, когда изменения происходят в определенные моменты времени, а результаты зависят от вероятностного распределения. При применении данного подхода переходы между состояниями происходят не непрерывно, а с некоторой вероятностью.

3.2. Определение ограничений, накладываемых областью применимости этого подхода

Данный подход применим к системам, в которых переходы подвержены случайным влияниям, а также дискретно-стохастический подход накладывает ряд ограничений:

- Для систем с большим количеством состояний данный подход плохо применим, так как моделирование систем становится сложно-управляемым;
- Дискретно-стохастический подход не применим для систем непрерывного характера, так как в данном подходе время и состояния представляют собой отдельные значения, определенные конкретным образом.
- Данных подход плохо применим для систем, обладающих сложной динамикой.

3.3. Определение цели моделирования

Цель моделирования абстрактной вычислительной системы при помощи дискретно-стохастического подхода – создание точной математической модели, которая необходима для анализа и прогнозирования работы системы в условиях распределения частоты запросов по заданным правилась аппроксимации.

3.4. Доопределение исходных данных, недостающих для применения этого подхода

Все необходимые данные, необходимые для моделирования остаются прежними. Для данного подхода важно более подробно рассмотреть гиперэкспоненциальное распределение.

Рассмотрим основные формулы, описывающие гиперэкспоненциальное распределение.

Дополнительная функция распределения:

$$\bar{F}(t) = \sum_{i=1}^{n} y_i e^{-\mu_i t}$$

Система уравнений относительно параметров аппроксимации:

$$y_1 + y_2 + \cdots + y_n = 1$$
$$\frac{y_1}{\mu_1} + \frac{y_2}{\mu_2} + \cdots + \frac{y_n}{\mu_n} = f_1$$
$$\frac{y_1}{\mu_1^2} + \frac{y_2}{\mu_2^2} + \cdots + \frac{y_n}{\mu_n^2} = \frac{f_2}{2}$$
$$\cdots$$
$$\frac{y_1}{\mu_1^i} + \frac{y_2}{\mu_2^i} + \cdots + \frac{y_n}{\mu_n^i} = \frac{f_i}{i!}$$
$$i = \overline{0,2n-1}$$

При проведении моделирования системы с использованием дискретно-стохастического подхода необходимо будет доработать модель, полученную в результате моделирования с использованием дискретно-детерминированного подхода, однако теперь необходимо более детально рассмотреть при помощи гиперэкспоненциального распределения время поступления запроса.

3.5. Проведение моделирования

При проведении моделирования, как уже было сказано ранее, нужно более подробно рассмотреть гиперэкспоненциальное распределение, которое будет определять, с какой периодичностью поступают запросы.

Гиперэкспоненциальное распределение контролирует моменты поступления запросов. Соответственно в момент поступления нового запроса время будет увеличиваться на $\Delta t_n = \frac{\ln U}{\lambda_i}$, где λ – интенсивность, U – случайная величина. Таким образом мы видим, что время поступления запроса определяется случайным образом.

После описания времени поступления запросов, необходимо дополнить систему. Дополненная система представлена на рисунке 7.

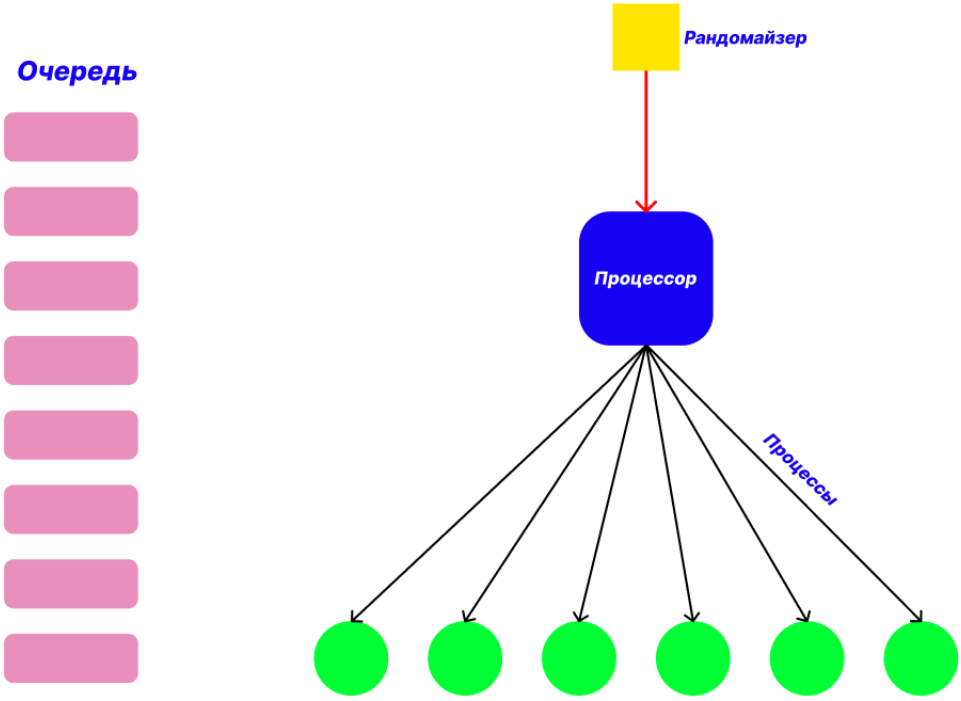

Рисунок 7 – Дополненная система.

Важно отметить, что состояния системы и переходы между состояниями остались прежними, однако теперь мы можем учитывать время поступления запросов, которое определено распределением.

3.6. Анализ результатов

Результаты применения дискретно-стохастического подхода к моделированию привели к значительному снижению неопределенности в отношении времени поступления запросов. Одним из ключевых преимуществ этого метода является его способность учитывать и моделировать случайные события и различные факторы неопределенности. Кроме того, дискретно-стохастическое моделирование отличается простотой формализации, что

позволяет создавать математически четкие и структурированные модели системы.

3.7. Выводы

Метод дискретно-стохастического моделирования показал себя как мощный инструмент для анализа моей системы. Этот подход не только позволяет оценить влияние приоритетов на общее функционирование системы, но и предоставляет возможность глубокого анализа ее поведения. Кроме того, дискретно-стохастическое моделирование дает возможность математически описать распределение времени поступления запросов в систему, что является критически важной характеристикой для оптимизации работы системы.

4. НЕПРЕРЫВНО-СТОХАСТИЧЕСКИЙ ПОДХОД

4.1. Краткое теоретическое описание

При непрерывно-стохастическом подходе в качестве типовых математических схем применяется система массового обслуживания (англ. queueing system), которые будем называть Q-схемами. Системы массового обслуживания представляют собой класс математических схем, разработанных в теории массового обслуживания и различных приложениях для формализации процессов функционирования систем, которые по своей сути являются процессами обслуживания.

В качестве процесса обслуживания могут быть представлены различные по своей физической природе процессы функционирования экономических, производственных, технических и других систем, например потоки поставок продукции некоторому предприятию, потоки деталей и комплектующих изделий на сборочном конвейере цеха, заявки на обработку информации ЭВМ от удаленных терминалов и т. д.

При этом характерным для работы таких объектов является случайное появление заявок (требований) на обслуживание и завершение обслуживания в случайные моменты времени, т. е. стохастический характер процесса их функционирования. Остановимся на основных понятиях массового обслуживания, необходимых для использования Q-схем, как при аналитическом, так и при имитационном.

В любом элементарном акте обслуживания можно выделить две основные составляющие:

ожидание обслуживания заявки;

собственно обслуживание заявки.

4.2. Определение ограничений, накладываемых областью применимости этого подхода

Непрерывно-стохастический подход является мощным инструментом в математической статистике и теории вероятностей, однако он имеет свои ограничения и области применимости. Вот некоторые из них:

- Линейность процессов: Непрерывно-стохастические модели часто предполагают линейные зависимости между переменными. Это может ограничивать их применение в ситуациях, где имеются значительные нелинейные отношения;
- Свойства случайных процессов: Часто предполагается, что случайные процессы являются стационарными или имеют определенные свойства (например, независимость или гауссовость). Если эти условия не выполняются, модель может неадекватно описывать реальность;
- Марковский характер процессов: Часто в непрерывно-стохастических моделях предполагается, что будущее состояния процесса зависит только от его текущего состояния (марковость). Это может не соответствовать действительности для многих процессов;
- Сложность вычислений: Моделирование и оценка параметров непрерывно-стохастических процессов может требовать значительных вычислительных ресурсов, особенно для сложных или многомерных моделей.

4.3. Определение цели моделирования

Главная цель моделирования системы с заданными параметрами путём применения непрерывно-стохастического подхода заключается в создании гибкой и адаптивной математической модели, которая позволяет не только анализировать и прогнозировать поведение системы или процесса в условиях неопределенности, но и обеспечивать оценку влияния различных факторов на динамику системы. Модель будет использовать непрерывные вероятностные процессы для оценки вероятностных характеристик системы, что позволяет глубже понять взаимодействия между компонентами системы и делать более обоснованные прогнозы.

4.4. Доопределение исходных данных, недостающих для применения этого подхода

Все необходимые данные для построения модели уже приведены.

При проведении моделирования с помощью непрерывно-стохастического подхода будет использоваться та же модель, что и для

дискретно-детерминированного подхода, но несколько доработанная. Отличие состоит в том, что теперь время отправления следующего запроса от каждого процесса к процессору представляет собой случайную величину, подчиняющуюся гиперэкспоненциального распределению.

4.5. Проведение моделирования

Для моделирования системы при помощи непрерывно-стохастического необходимо написать программу. Основные принципы работы системы описаны ранее. Время работы системы и данные для распределения будет менять для проведения анализа результатов моделирования. В процессе моделирования был написан код на языке python. Исходный код программы представлен в приложении к отчету.

4.6. Анализ результатов

Рассмотрим результаты моделирования системы. Первый вариант исходных данных:

- λ_1 = 60
- λ_2 = 10
- t = 100

График занятости очереди и процессов № 1 представлен на рисунке 8.

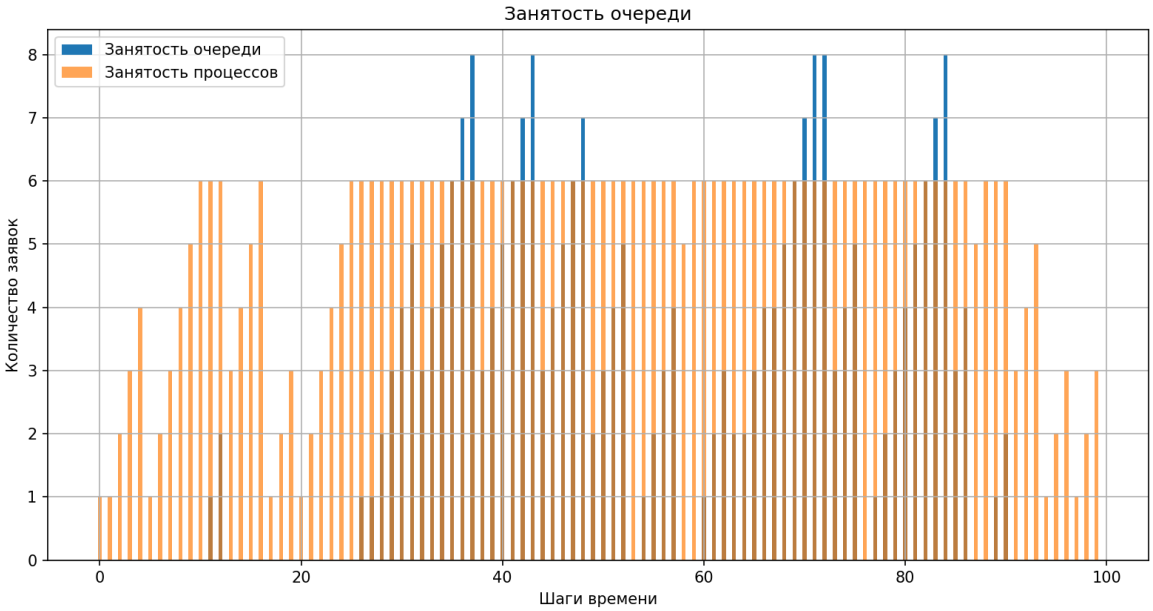

Рисунок 8 – График занятости очереди и процессов № 1.

Результат моделирования № 1 представлен на рисунке 9.

```
СТАТИСТИКА:
Общее число заявок: 100
Обработано заявок: 96
Отброшено заявок: 0
Общее время ожидания: 10.218 сек
Среднее время ожидания: 0.106 сек
Суммарное время занятости процессора: 13.600 сек
Эффективность системы: 96.00%
```

Рисунок 9 – Результат моделирования № 1.

Второй вариант исходных данных:

- λ_1 = 100
- λ_2 = 5
- t = 10

График занятости очереди и процессов № 2 представлен на рисунке 10.

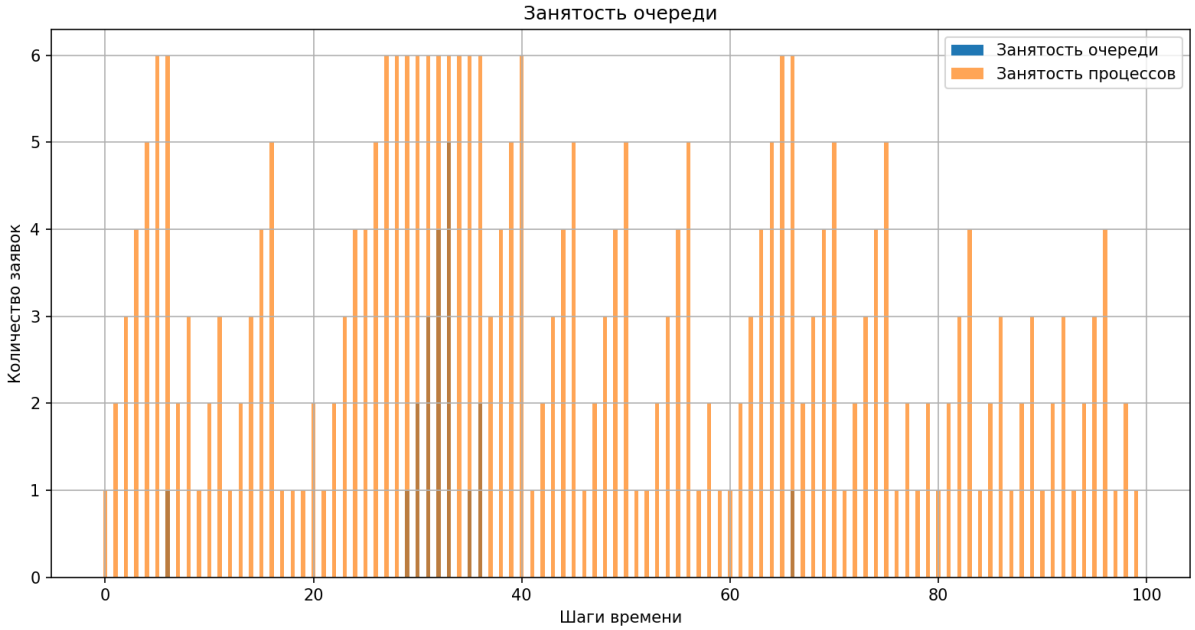

Рисунок 10 – График занятости очереди и процессов № 2.

Результат моделирования № 2 представлен на рисунке 11.

```
СТАТИСТИКА:
Общее число заявок: 100
Обработано заявок: 99
Отброшено заявок: 0
Общее время ожидания: 2.438 сек
Среднее время ожидания: 0.025 сек
Суммарное время занятости процессора: 13.600 сек
Эффективность системы: 99.00%
```

Рисунок 11 – Результат моделирования № 2.

Третий вариант исходных данных:

- λ_1 = 10
- λ_2 = 50
- t = 200

График занятости очереди и процессов № 3 представлен на рисунке 12.

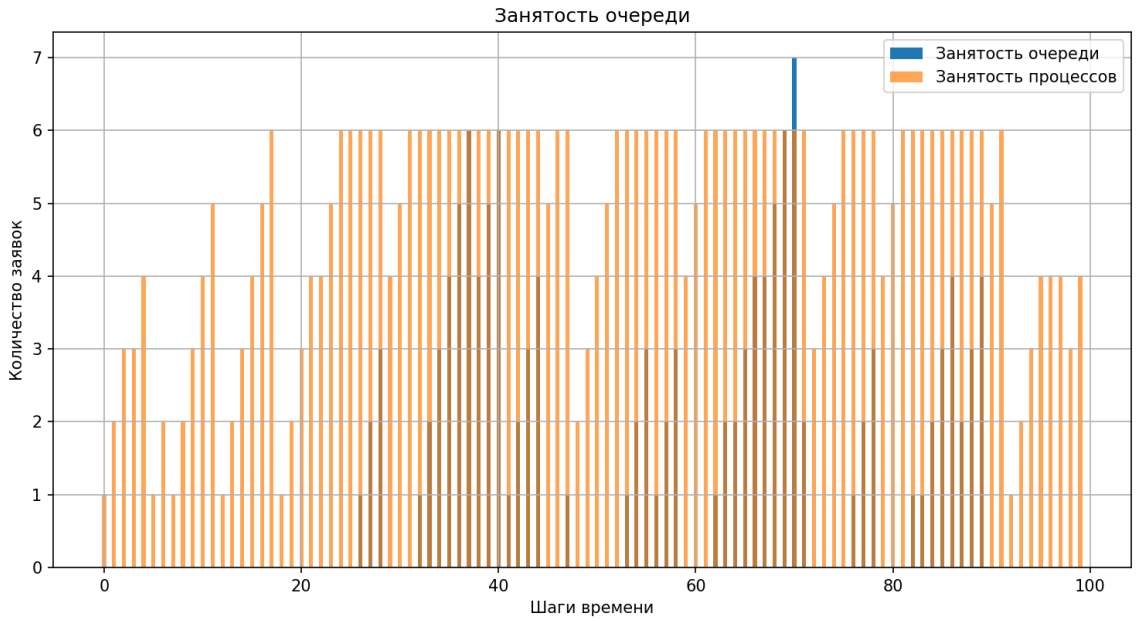

Рисунок 12 – График занятости очереди и процессов № 3.

Результат моделирования № 3 представлен на рисунке 13.

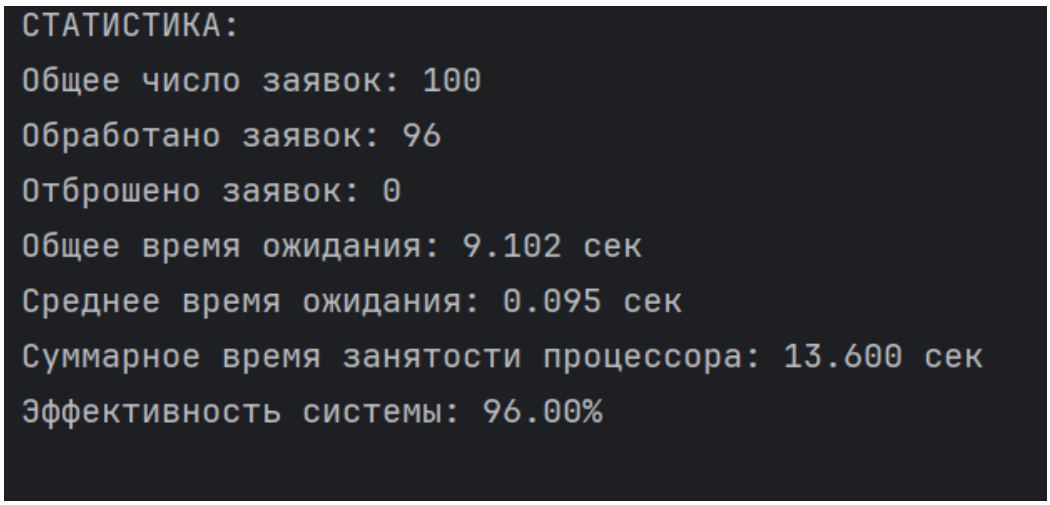

Рисунок 13 – Результат моделирования № 3.

4.7. Выводы

В результате моделирования можно сделать вывод, что непрерывно-стохастический подход отлично подходит для моделирования моей системы, так как заявки обрабатываются достаточно быстро, очередь почти не собирается и системы показывает высокую эффективность обработки заявок.

5. СЕТЕВОЙ ПОДХОД

5.1. Краткое теоретическое описание

Сетевой подход применяется в тех случаях, когда необходимо изучить и проанализировать сложную систему. Его основной принцип заключается в построении графических моделей, которые отображают логические связи между элементами системы.

Сетевой подход используется для анализа систем путем построения графа, в котором вершины описывают состояния системы, которые были определены ранее, а ребра – переходы между состояниями.

Данный метод позволяет формализовать работу системы, оценить вероятности состояний и ключевые метрики, провести моделирование стохастических событий.

5.2. Определение ограничений, накладываемых областью применимости этого подхода

Сетевой подход применим для систем, где:

- Состояния четко определены, то есть известны состояния системы, возможные переходы между состояниями, приоритеты запросов.
- Также должны быть возможность описать события правилами переходов.
- Известно распределение событий, в моем случае появление запросов в системе описывается гиперэкспоненциальным распределением.
- Также должен быть ограниченный размер очередь.

Все эти ограничения делают подход плохо-применимым для сложных систем с большим количеством состояний.

5.3. Определение цели моделирования

Основной целью моделирования абстрактной вычислительной системы при помощи сетевого подхода является построение формальной модели работы системы, которая отражает обработку запросов, управление очередью на основании приоритетов и принцип отклонения или вытеснения запросов. Также к целям моделирования системы при помощи сетевого подхода можно

отнести оценку вероятности и проведение оптимизации управления системой, то есть провести проверку того, насколько текущие правила работы эффективны.

5.4. Доопределение исходных данных, недостающих для применения этого подхода

Все необходимые исходные данные, необходимые для проведения моделирования, были определены ранее и остаются неизменными.

5.5. Проведение моделирования

Места:

- Поступление запроса;
- Свободный поток;
- Выход запроса;
- Отказ.

Переходы:

- Буфер;
- Обработка;
- Буфер переполнен.

Создадим граф, который будет визуализировать места и переходы. Места и переходы представлены на рисунке 14.

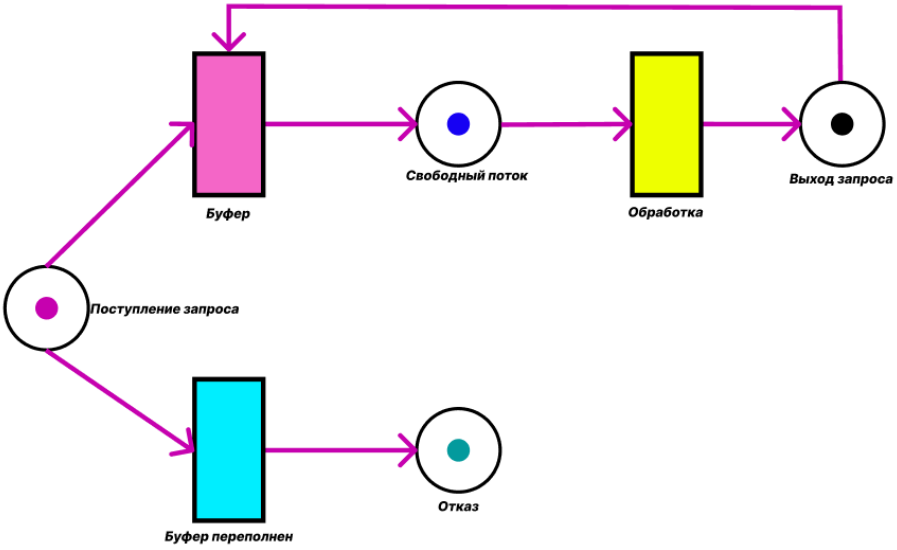

Рисунок 14 – Места и переходы.

5.6. Анализ результатов

В результате моделирования была получена визуализация работы системы при помощи графа. Можно сделать вывод, что сетевой подход отлично подходит для систем, которые имеют небольшое количество состояний. Также сетевой подход необходим для визуализации работы системы.

5.7. Выводы

Сетевой подход отлично подходит для моделирования абстрактных вычислительных систем, которые имеют небольшое количество состояний, а также помогает в визуализации системы для анализа принципа работы системы и внесения изменений в структуру работы, если это необходимо.

6. ОБОБЩЕННЫЙ ПОДХОД

6.1. Краткое теоретическое описание

Обобщенный подход к моделированию абстрактной вычислительной системы необходим для описания системы используя математические модели. Подход используется для исследования основных свойства и поведения системы. К основным этапам данного подхода можно отнести:

- Формализацию системы, то есть определить ее параметры, состояния и события;
- Математическое описание поведения системы;
- Анализ системы и проверка гипотез.

6.2. Определение ограничений, накладываемых областью применимости этого подхода

Обобщенный подход имеет ряд ограничений относительно систем, к которым он может быть применим, а именно:

- Подход плохо применим для систем с высокой сложностью событий, так как могут потребоваться ресурсоемкие вычисления;
- Системы с нелинейным поведением плохо описываются обобщенным подходом;
- При данном подходе описание ограничивается общими свойствами, то есть, когда в системе есть какие-то универсальные события, это сложно учесть.

6.3. Определение цели моделирования

Основные цели моделирования абстрактной вычислительной системы с использованием обобщенного подхода – это анализ производительности и оптимизация параметров. То есть в результате моделирования можно увидеть несовершенства системы, а также лучше изучить поведение системы и, соответственно, оценить ее производительность.

6.4. Доопределение исходных данных, недостающих для применения этого подхода

Все необходимые данные были определены ранее, однако добавим, что время, в течение которого мы будем рассматривать работу системы равняется 10 секундам. Остальные параметры остаются прежними.

6.5. Проведение моделирования

Определим обозначения для схемы:

- Z – текущий запрос;
- p – приоритет текущего запроса;
- p0, p1, p2, p3, p4, p5, p6 – приоритеты запросов на позициях буфера;
- t – текущее время;
- delta t – время, определенное гиперэкспоненциальным распределением.

На рисунке 15 представлена схема работы системы.

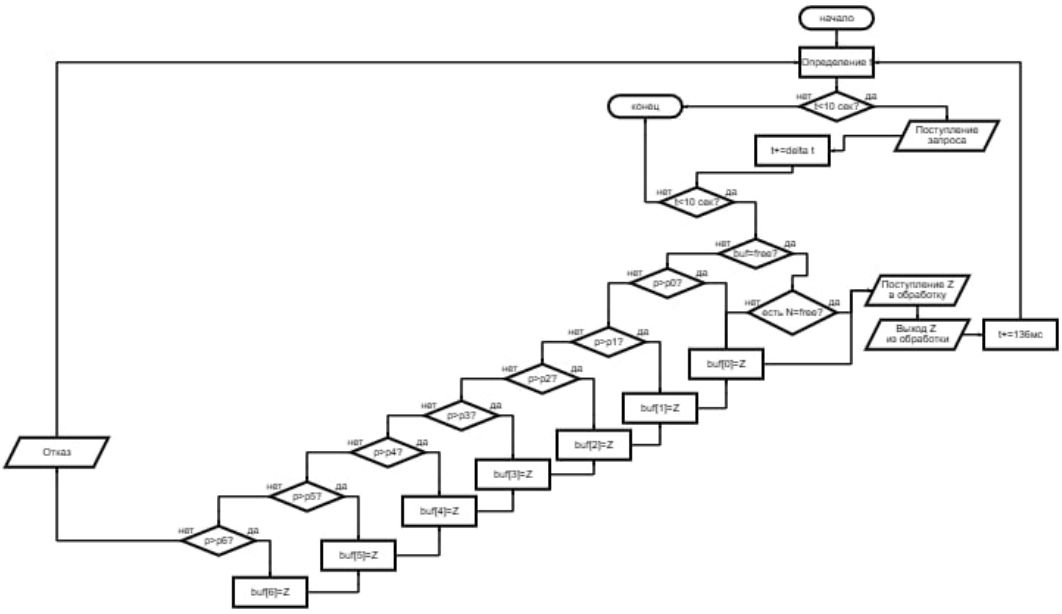

Рисунок 15 – Схема работы системы.

6.6. Анализ результатов

В результате была получена схема, которая отлично визуализирует весь цикл работы системы. В схеме представлено, как именно пришедший запрос обрабатывается системой.

6.7. Выводы

Обобщенный подход хорошо применим для моделирования систем, однако важно, чтобы исходные данные были достаточно полноценные, а также система не была слишком сложной, потому что, в первом варианте, моделирование невозможно, а во втором случае, схема получится слишком сложной, запутанной и анализ результатов будет невозможен.

ЗАКЛЮЧЕНИЕ

Кратко подвести итоги, проанализировать соответствие поставленной цели и полученного результата.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

Ниже представлены примеры библиографического описания, В КАЧЕСТВЕ НАЗВАНИЯ ИСТОЧНИКА в примерах приводится вариант, в котором применяется то или иное библиографическое описание.

1. Иванов И. И. Книга одного-трех авторов. М.: Издательство, 2010. 000 с.

2. Книга четырех авторов / И. И. Иванов, П. П. Петров, С. С. Сидоров, В. В. Васильев. СПб.: Издательство, 2010. 000 с.

3. Книга пяти и более авторов / И. И. Иванов, П. П. Петров, С. С. Сидоров и др.. СПб.: Издательство, 2010. 000 с.

4. Описание книги под редакцией / под ред. И.И. Иванова СПб., Издательство, 2010. 000 с.

5. Иванов И.И. Описание учебного пособия и текста лекций: учеб. пособие. СПб.: Изд-во СПбГЭТУ «ЛЭТИ», 2010. 000 с.

6. Описание методических указаний / сост.: И.И. Иванов, П.П. Петров. СПб.: Изд-во СПбГЭТУ «ЛЭТИ», 2010. 000 с.

7. Иванов И.И. Описание статьи с одним-тремя авторами из журнала // Название журнала. 2010, вып. (№) 00. С. 000–000.

8. Описание статьи с четырьмя и более авторами из журнала / И. И. Иванов, П. П. Петров, С. С. Сидоров и др. // Название журнала. 2010, вып. (№) 00. С. 000–000.

9. Иванов И.И. Описание тезисов доклада с одним-тремя авторами / Название конференции: тез. докл. III международной науч.-техн. конф., СПб, 00–00 янв. 2000 г. / СПбГЭТУ «ЛЭТИ», СПБ, 2010, С. 000–000.

10. Описание тезисов доклада с четырьмя и более авторами / И. И. Иванов, П. П. Петров, С. С. Сидоров и др. // Название конференции: тез. докл. III международной науч.-техн. конф., СПб, 00–00 янв. 2000 г. / СПбГЭТУ «ЛЭТИ», СПБ, 2010, С. 000–000.

11. Описание электронного ресурса // Наименование сайта. URL: http://east-front.narod.ru/memo/latchford.htm (дата обращения: 00.00.2010).

12. ГОСТ 0.0–00. Описание стандартов. М.: Изд-во стандартов, 2010.

13. Пат. RU 00000000. Описание патентных документов / И. И. Иванов, П. П. Петров, С. С. Сидоров. Опубл. 00.00.2010. Бюл. № 00.

14. Иванов И.И. Описание авторефератов диссертаций: автореф. дисс. канд. техн. наук / СПбГЭТУ «ЛЭТИ», СПБ, 2010.

15. Описание федерального закона: Федер. закон [принят Гос. Думой 00.00.2010] // Собрание законодательств РФ. 2010. № 00. Ст. 00. С. 000–000.

16. Описание федерального постановления: постановление Правительства Рос. Федерации от 00.00.2010 № 00000 // Опубликовавшее издание. 2010. № 0. С. 000–000.

17. Описание указа: указ Президента РФ от 00.00.2010 № 00 // Опубликовавшее издание. 2010. № 0. С. 000–000.

ПРИЛОЖЕНИЕ 1
ИСХОДНЫЙ КОД ПРОГРАММЫ

```
import random
import numpy as np
import heapq
import matplotlib.pyplot as plt

# Константы
PROCESSING_TIME = 0.136  # Время обработки одной заявки
(сек)
LAMBDA_1 = 10          # Параметр
гиперэкспоненциального распределения
LAMBDA_2 = 50
P_LAMBDA = [0.5, 0.5]   # Вероятности выбора
распределения
QUEUE_SIZE = 8          # Максимальный размер очереди
MAX_PROCESSES = 6       # Количество одновременно
работающих процессов
PRIORITY_LEVELS = 6     # Уровни приоритетов

# Генерация времени через гиперэкспоненциальное
распределение
def generate_hyperexponential():
    selected_lambda = np.random.choice([LAMBDA_1,
LAMBDA_2], p=P_LAMBDA)
    return np.random.exponential(1 / selected_lambda)

# Генерация случайного приоритета
def generate_priority():
    return random.randint(1, PRIORITY_LEVELS)
```

```
# Класс заявки
class Request:
    def __init__(self, arrival_time, priority):
        self.arrival_time = arrival_time
        self.priority = priority
        self.start_time = None
        self.finish_time = None

    def __lt__(self, other):
        return self.priority > other.priority  # Высший
приоритет ближе к началу очереди

# Класс системы
class System:
    def __init__(self):
        self.current_time = 0
        self.processor = []  # Занятые процессы
        self.queue = []      # Очередь заявок
        self.total_requests = 0
        self.processed_requests = 0
        self.dropped_requests = 0
        self.total_wait_time = 0
        self.busy_time = 0
        self.buffer_usage = []
        self.process_usage = []

    def process_request(self, request):
        """Начало обработки заявки"""
        request.start_time = self.current_time
```

```
        request.finish_time = self.current_time +
PROCESSING_TIME
        heapq.heappush(self.processor, request)

    def step(self, new_request=None):
        """Шаг моделирования"""
        # Завершаем обработку заявок
        while self.processor and
self.processor[0].finish_time <= self.current_time:
            finished_request =
heapq.heappop(self.processor)
            self.processed_requests += 1
            self.total_wait_time +=
finished_request.start_time -
finished_request.arrival_time

        # Добавляем новую заявку
        if new_request:
            if len(self.processor) < MAX_PROCESSES:
                self.process_request(new_request)
            elif len(self.queue) < QUEUE_SIZE:
                heapq.heappush(self.queue, new_request)
            else:
                # Если очередь заполнена, проверяем
приоритет
                lowest_priority_request = self.queue[-
1]
                if new_request.priority >
lowest_priority_request.priority:
                    heapq.heappop(self.queue)
```

```
                    heapq.heappush(self.queue,
new_request)
                else:
                    self.dropped_requests += 1

        # Перемещаем заявки из очереди в процессор
        while len(self.processor) < MAX_PROCESSES and
self.queue:
            next_request = heapq.heappop(self.queue)
            self.process_request(next_request)

        # Сохраняем данные о занятости ресурсов
        self.buffer_usage.append(len(self.queue))
        self.process_usage.append(len(self.processor))

        # Учитываем занятость процессора
        if self.processor:
            self.busy_time += PROCESSING_TIME

    def run(self, max_time=10, max_requests=100000000):
        """Запуск моделирования"""
        while self.current_time < max_time and
self.total_requests < max_requests:
            # Генерация новой заявки
            interarrival_time =
generate_hyperexponential()
            self.current_time += interarrival_time
            new_request = Request(self.current_time,
generate_priority())
            self.total_requests += 1
```

```
            self.step(new_request)

        # Расчёт метрик
        efficiency = self.processed_requests /
self.total_requests if self.total_requests > 0 else 0
        avg_wait_time = self.total_wait_time /
self.processed_requests if self.processed_requests > 0
else 0

        return {
            "total_requests": self.total_requests,
            "processed_requests":
self.processed_requests,"dropped_requests":
self.dropped_requests,
            "total_wait_time": self.total_wait_time,
            "avg_wait_time": avg_wait_time,
            "busy_time": self.busy_time,
            "efficiency": efficiency,
            "buffer_usage": self.buffer_usage,
            "process_usage": self.process_usage,
        }

# Основной код
if __name__ == "__main__":
    system = System()
    stats = system.run(max_time=10, max_requests=100)

    # Вывод статистики
    print("СТАТИСТИКА:")
    print(f"Общее число заявок:
```

```
{stats['total_requests']}")
    print(f"Обработано заявок:
{stats['processed_requests']}")
    print(f"Отброшено заявок:
{stats['dropped_requests']}")
    print(f"Общее время ожидания:
{stats['total_wait_time']:.3f} сек")
    print(f"Среднее время ожидания:
{stats['avg_wait_time']:.3f} сек")
    print(f"Суммарное время занятости процессора:
{stats['busy_time']:.3f} сек")
    print(f"Эффективность системы:
{stats['efficiency']:.2%}")

    # График занятости
    time_points = range(len(stats['buffer_usage']))
    plt.figure(figsize=(12, 6))
    plt.bar(time_points, stats['buffer_usage'],
width=0.4, label="Занятость очереди")
    plt.bar(time_points, stats['process_usage'],
width=0.4, label="Занятость процессов", alpha=0.7)
    plt.title("Занятость очереди")
    plt.xlabel("Шаги времени")
    plt.ylabel("Количество заявок")
    plt.legend()
    plt.grid()
    plt.show()
```