**1. введение**

Моделирование производительности систем параллельной обработки данных должно учитывать структуру заданий и саму

структуру системы. Структура задания описывает

отношения приоритета между различными компонентами задания, называемыми

задачами, и обычно задается графовой

моделью задачи. Простая структура задания - это разветвление / объединение, при котором

задание состоит из набора независимых задач. Для такой

структуры задание считается выполненным только тогда, когда выполнены все

его задачи. Структура системы

моделируется системой массового обслуживания, состоящей из системы очередей

и набор серверов. Вообще говоря, у нас может быть

либо центральная очередь, доступная всем процессорам

, либо распределенная очередь, где каждый процессор обслуживает

свою собственную очередь. Очередь может содержать задания или задачки в зависимости

от того, разделены ли задания на задачи перед входом в

очередь или нет. Мы индексируем задачи, составляющие задание, по

порядку их выполнения и определяем время между

выполнением первой и последней задачи как задержку синхронизации

этого задания. Время отклика задания, которое включает

в себя эту задержку синхронизации, трудно получить даже для простого

системные структуры [ 13, [ 3 ] - [ 5 ] . В неразделяющемся случае анализ намного проще, поскольку все задачи из одного и того же задания выполняются последовательно на одном и том же процессоре.

Шпионы и Гейленкейзер [6] изучают родственную систему. Они

рассматривают несколько процессоров, обслуживающих задания, которые поступают в соответствии

с пуассоновским процессом. Каждое задание изначально состоит

из одной задачи. По завершении каждая задача генерирует

ноль или более дополнительных задач, требующих дальнейшей обработки.

Следовательно, задание представлено древовидным графом приоритета

, где управление передается от корня к листьям. Статистика

количества задач в системе выводится

в предположении, что время обслуживания задач является независимой

и одинаково распределенной экспоненциальной случайной величиной.

Однако о результатах, касающихся

характеристик времени отклика задания, не сообщается.

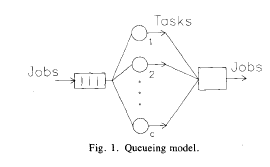
В этой статье мы моделируем систему параллельной обработки с центральной очередью и разделением заданий как систему массового поступления Mx / M / c, в которой клиенты и объемы соответствуют задачам и заданиям, соответственно. Такая система была изучена в работах [2], [7], и получено выражение для среднего времени отклика случайного клиента. Однако, поскольку нас интересует время, которое задание проводит в системе, включая задержку синхронизации, мы должны оценить массовое время отклика, а не просто время отклика клиента.

В разделе II мы опишем рассматриваемую систему массового обслуживания. Время отклика на задание - это сумма времени ожидания задания и времени обслуживания задания. В разделе III мы анализируем систему массового обслуживания и получаем выражение для среднего времени ожидания задания. Среднее время выполнения задания определяется набором рекуррентных уравнений. В разделе IV мы рассматриваем модели массового обслуживания широкого спектра систем параллельной обработки, охватывающих системы с централизованными и распределенными очередями, а также с разделением заданий и без него. В этих моделях мы предполагаем, что задания состоят из графов задач разветвления /объединения, в которых каждая задача имеет одинаковое распределение обслуживания, которое предполагается экспоненциальным. Это упрощающие допущения, и они сделаны для того, чтобы результирующие системы массового обслуживания были математически управляемыми. В реальных системах задания могут состоять из произвольного графа задач и могут иметь очень разные распределения служб задач. Однако мы считаем, что фундаментальные характеристики производительности отражены в наших моделях и что взаимосвязи, найденные путем сравнения их относительного времени отклика в разделе V, по-прежнему сохраняются при менее строгих допущениях. Другие характеристики, которые мы изучаем в разделе V, включают влияние параллелизма на время отклика и накладные задержки из-за разделения заданий.

**2. ОПИСАНИЕ МОДЕЛИ**

Мы рассматриваем систему из c идентичных серверов, обслуживающих одну очередь. Задания поступают в систему в соответствии с пуассоновским процессом с параметром lambda. Каждое задание состоит из одной или нескольких задач, которые могут обрабатываться независимо друг от друга. В частности, пусть X - **случайная величина** (r.v.), которая обозначает количество задач в задании. Мы предполагаем, что X имеет распределение вероятностей ai = P [ X = i 1, i = 1 , 2 .... и **функцию генерации вероятности** (p.g.f.) X ( z ) = Er = qz'. Предполагается, что время обслуживания, требуемое для выполнения задачи, является экспоненциальным r.v. с параметром mu и не зависит от требований к обслуживанию всех других задач.

Нас интересует устойчивое поведение этой системы. В частности, мы фокусируемся на времени отклика случайного задания, т.е. на интервале времени, измеряемом с момента поступления задания до завершения обслуживания последней задачи, связанной с этим заданием. Можно представить систему как состоящую из очереди для задач, ожидающих обслуживания, серверов C и зоны ожидания для задач, которые завершили обслуживание, но ожидают завершения последней задачи, связанной с заданием (рис. 1). Все задачи в рамках задания покидают эту последнюю зону ожидания, когда все они завершат свое обслуживание. Обозначим это время отклика как T. Наконец, мы предполагаем, что задания планируются к выполнению в порядке живой очереди. Задачи в рамках одного задания планируются в случайном порядке.



**3. АНАЛИЗ**

Как описано, эта система моделируется как

марковский процесс с непрерывным временем и дискретным состоянием. Мы получаем статистику

времени отклика задания, разбивая

проблему на две части, каждая из которых может быть обработана

простым способом. В частности, время отклика T случайного

задания может быть выражено как сумма двух слагаемых: T =

W + S. Первый член W - это время ожидания задания и соответствует

времени, в течение которого задание ожидает в очереди, прежде

чем будет запланирована первая из его задач. Если задание поступает в

систему, где один или несколько серверов простаивают, то когда

= = 0, в противном случае W > 0. Второе условие

- это время обслуживания задания и соответствует времени, необходимому для обработки

всех задач, связанных с заданием, после того, как запланирована первая

задача. Среднее время отклика определяется по формуле



Статистику для W можно получить, изучив очередь массового

поступления M"/M/c, которая лежит

в основе интересующей нас системы параллельной обработки. В частности, в нашей

системе это соответствует времени, которое основная масса клиентов

в M"/M/c должна ждать, прежде чем первый клиент начнет

обслуживание. Раздел 111-А посвящен этой случайной величине.

Статистика времени выполнения задания приведена в разделе

111-Б. К сожалению, мы не можем получить

выражения замкнутой формы для этих статистических данных, но должны численно

решить набор рекуррентных соотношений.

1. **Среднее время ожидания задания**

В этом разделе мы анализируем поведение очереди M x / M / c

. Это дает нам статистику для W, времени

ожидания задания в очереди системы параллельной обработки

. Распределение длины очереди для массового поступления

Очередь M " / M / c также будет использоваться для получения

статистики времени обслуживания в системе параллельной обработки. Мы будем

использовать следующую терминологию. Система массового поступления

обрабатывает задачи. Задачи, поступающие одновременно, образуют

группу задач. Читатель должен отметить, что задание в системе

параллельной обработки соответствует группе задач

в системе массового прибытия. В дополнение к обозначениям, введенным

в предыдущем разделе, мы определяем N как случайную

величину, обозначающую установившееся количество задач в

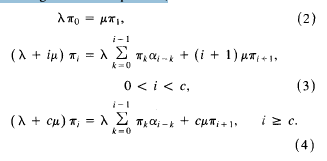
системе. N имеет распределение ….



Установившееся состояние

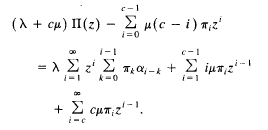
распределение длины очереди, пи\_i, i = 0, 1, - удовлетворяет

следующим разностным уравнениям,



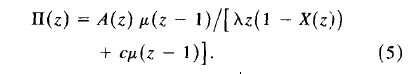
Умножая обе части каждого из приведенных выше уравнений

на zi и суммируя, мы получаем



Определив  и подставив это

в приведенное выше уравнение, мы, наконец, получим



Принимая предел приведенного выше уравнения за z, равный 1,

мы получаем

который как   даеlftnт



Приведенное выше

выражение вместе с (2) и (3) может быть использовано для получения

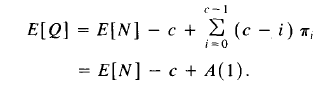
выражения для П(z). Свойства

p.g.f., генерирующие момент, позволяют нам получить ожидаемое количество

задач в системе, т.е 

Пусть Q - значение случайно велечины, обозначающее количество задач в очереди

, ожидающих обслуживания; тогда мы имеем



\ne



На этом мы завершаем наше обсуждение массового прибытия

Очередь M"/M/c.

1. ***Mean Job Service Time***

В этом разделе мы определяем значение среднего

времени выполнения задания E[ S ] для случайного задания. Анализ упрощается

, если мы зависим от времени обслуживания задания, от количества

процессоров, которые могут быть первоначально запланированы (т.е. на

момент планирования первой задачи), и от общего количества

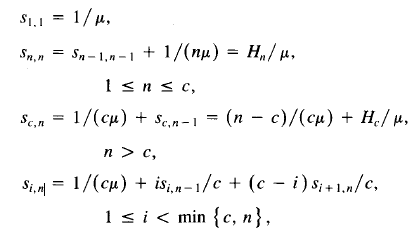
задач, которые должны быть выполнены (включая те, которые находятся в

обслуживании).

Пусть  среднее время обслуживания задания, учитывая, что необходимо

выполнить n задач и что i задач изначально запланированы для серверов.

Величина s\_i\_n удовлетворяет следующим уравнениям,



где  является гармоническим рядом

Количество серверов, доступных при первом

планировании задания, связано с количеством незанятых серверов на

момент поступления задания. Если задание поступает, когда все серверы заняты,

N >= c, тогда при

планировании первой задачи будет доступен ровно один сервер. Если на момент поступления задания имеется один или несколько незанятых серверов

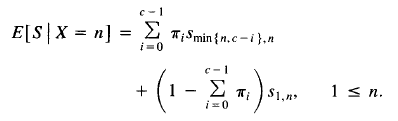
, N < c, то задание

запланировано немедленно и первоначально на это время будет запланировано минимальное количество { X,

c-N } серверов. Следовательно, устранение

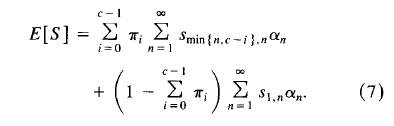
ограничений на количество изначально доступных серверов

приводит к



Устранение зависимости от количества задач в

задании приводит к



Среднее время отклика задания E [ T] получено путем подстановки

(6) и (7) в ( 1 ) .

В качестве примера мы рассмотрим частный случай, когда

количество задач в задании является случайно геометрической величиной, т.е. P [ X = n ]  Мы сосредоточимся на определении

ожидаемого времени обслуживания, например E[S]. Следующий анализ

стал возможен благодаря свойству геометрического

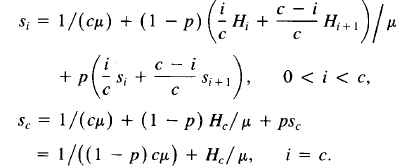
распределения без памяти. Пусть s\_i, 0 < i<= c - среднее

время обслуживания задания, учитывая, что i задач изначально запланированы для i

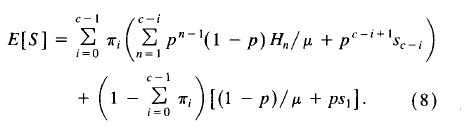
процессоров и что оно содержит по крайней мере одну задачу в

очереди. Это условное ожидание удовлетворяет следующим

уравнениям,



Путем подстановки в (7) получаем



**IV. МОДЕЛИ ПАРАЛЛЕЛЬНОЙ ОБРАБОТКИ**

В оставшейся части этой статьи мы рассмотрим следующее

вопрос. Как следует спроектировать систему, которая обрабатывает

поступающий поток заданий, состоящий из независимых

задач, в системе, состоящей из c идентичных процессоров?

Поскольку система массового обслуживания, проанализированная в предыдущем разделе

, является лишь одним из способов построения системы параллельной обработки,

мы считаем, что в этом разделе больший интерес

представляет расширение класса архитектур, которые мы рассматриваем.

На рис. 2 мы представляем четыре модели систем параллельной обработки

: распределенная/с разделением (D/S), распределенная/без разделения

(D/NS), централизованный/разделенный (C/S) и централизованный/

никакого разделения (C/NS). В каждой из этих систем есть

процессоры c, предполагается, что задания состоят из набора задач, которые

независимы и имеют экспоненциально распределенные

требования к обслуживанию, и предполагается, что поступления заданий

поступают из точечного источника Пуассона с интенсивностью A.

Системы различаются способом постановки заданий в очередь для процессоров

и способом планирования заданий на процессорах. То

постановка заданий в очередь для процессоров распределяется, если у каждого процессора

есть своя собственная очередь, и централизуется, если существует

общая очередь для всех процессоров. Планирование

заданий на процессорах не является разделением, если весь набор

задач, составляющих это задание, запланирован для последовательного выполнения

на одном и том же процессоре после того, как задание запланировано. С

другой стороны, планирование разделяется, если задачи задания

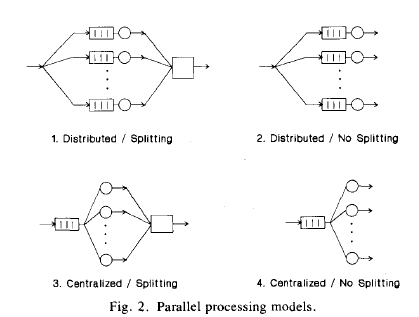
запланированы таким образом, что они могут выполняться независимо и

потенциально параллельно на разных процессорах.

В случае разделения задание завершается только тогда, когда все его задачи завершили выполнение. Это известно как операция объединения. Мы

выделяем этот случай на рисунках, показывая поле “ожидание”

после процессоров в системах ./S.



В нашем исследовании мы сравниваем среднее время отклика заданий

в каждой из систем для различных значений количества

процессоров, количества задач на задание, загрузки сервера

и определенных накладных расходов, связанных с разделением задания.

Система M ” / M /c, рассмотренная в предыдущем разделе, соответствует

системе C/S. В этой системе, когда процессоры

освобождаются, они обслуживают первую задачу в очереди. D/.

системы изучаются в работе [5]. Мы используем приблизительный анализ

системы D/S и точный анализ системы DINS

, которые приведены в этой статье. Для систем с 32

процессоров или меньше, было обнаружено, что относительная погрешность аппроксимации

для системы D/S составляет менее 5 процентов.

Приближение для системы D/S, приведенное в [5], требует

, чтобы количество задач на задание равнялось количеству

процессоров, каждая задача запланирована на другом процессоре.

В системе D/NS задания назначаются процессорам

с равной вероятностью. Приближение, которое мы используем

для среднего времени отклика задания для системы C/NS

, найдено в [8]. Хотя подробный анализ ошибок для этой

системы во всех диапазонах параметров не проводился,

наибольшая относительная погрешность для системы M / E\_2/10, приведенная

в [8], составляет около 0,1 процента.

**5. Результат**