ДЕТЕРМИНИРОВАННАЯ МОДЕЛЬ ПАКЕТНОЙ ОБРАБОТКИ ИНФОРМАЦИОННЫХ ЗАПРОСОВ В МНОГОПОТОЧНОМ СЕРВЕРЕ

Асратян Р.Э.
Институт проблем управления им. В.А. Трапезникова РАН, Москва, Россия
rubezas@yandex.ru

Аннотация. Рассматривается аналитическая модель обработки пакета однотипных запросов в однопроцессорном многопоточном сервере, основанная на двумерной системе рекуррентных соотношений, связывающих времена начала выполнения отдельных этапов обработки каждого запроса в пакете с временами начала выполнения отдельных этапов обработки других запросов. Показано, что модель позволяет связать основные характеристик обрабатывающего программного сервера с характеристиками пакета запросов и получить аналитические результаты, имеющие значения для практики создания распределенных систем.

Ключевые слова: распределенные системы, детерминированная модель, пакетный запрос, пакетная обработка, многопоточный сервер.

Введение

В данной работе под пакетной обработкой понимается группировка одиночных информационных запросов в пакеты для повышения производительности. Возможность обработки целого массива запросов, включающего десятки или сотни элементов, за одно обращение к серверу как правило позволяет клиенту многократно увеличить количество запросов, обрабатываемых, например, в течение суток [1, 2]. Разумеется, данный подход требует разработки специальных сервисов в составе распределенных систем, способных эффективно обрабатывать массивы запросов и формировать соответствующие им массивы «ответов», содержащие результаты обращения.

Работа таких сервисов как правило опирается на аппарат параллельных программных нитей (потоков), имеющийся в составе современных операционных систем [3-5]. Путем создания множества программных нитей для обработки множества запросов в пакете сервис получает возможность значительно повысить свою производительность (число обработанных запросов в единицу времени) даже в однопроцессорном сервере [3, 6]. Поскольку логика обработки каждого запроса как правило включает длительные по времени обращения к серверам СУБД, порождающие длительные периодами ожидания в работе программных нитей, пакетная обработка дает возможность резко сократить простои процессора по сравнению с последовательной обработкой [3, 7] и радикально повысить производительность. Тем не менее, в литературе практически не упоминаются исследования пакетных сервисов, позволяющие связать выигрыш в производительности с характеристиками пакета запросов.

Некоторое время назад автору пришлось принять участие в экспериментальной оценке возможностей многопоточного программного сервера приложений, включающего средства пакетной обработки запросов. Обработка каждого запроса в пакете включала всего три основных фазы: разбор параметров запроса, обращение к серверу СУБД для выборки релевантной информации и оформление результата выборки в формате XML-документа. Оказалось, что даже в относительно простом случае объяснить связь между длительностями этих трех фаз и временем обработки всего пакета достаточно непросто. Например, когда был найден способ вдвое сократить время выборки данных из БД оказалось, что в пакетах малого размера скорость обработки запросов значительно выросла, а в больших пакетах почему-то практически не изменилась. В частности, предельная производительность сервера на пакетах большого размера в обоих случаях оказалась приблизительно одинаковой (см. рис. 1). В целом проведенная оценка привела к следующим основным выводам:

- пакетная обработка позволяет в несколько раз повысить производительность сервера и может оказаться единственным возможным решением, позволяющим справиться с высокой нагрузкой;
- использование аппарата параллельных программных нитей, разделяющих общее адресное пространство (т.е. «многопоточности»), гораздо более эффективно для поддержки пакетной обработки запросов, чем использование аппарата параллельных процессов («многопроцессности»), так как связан со значительно меньшими «накладными расходами»;
- создание аналитической модели пакетной обработки является актуальной задачей, т.к. без нее невозможно обосновать или предсказать даже качественное «поведение» сервера, а тем более – связать количественные характеристики его быстродействия с характеристиками запросов и обрабатывающих нитей.

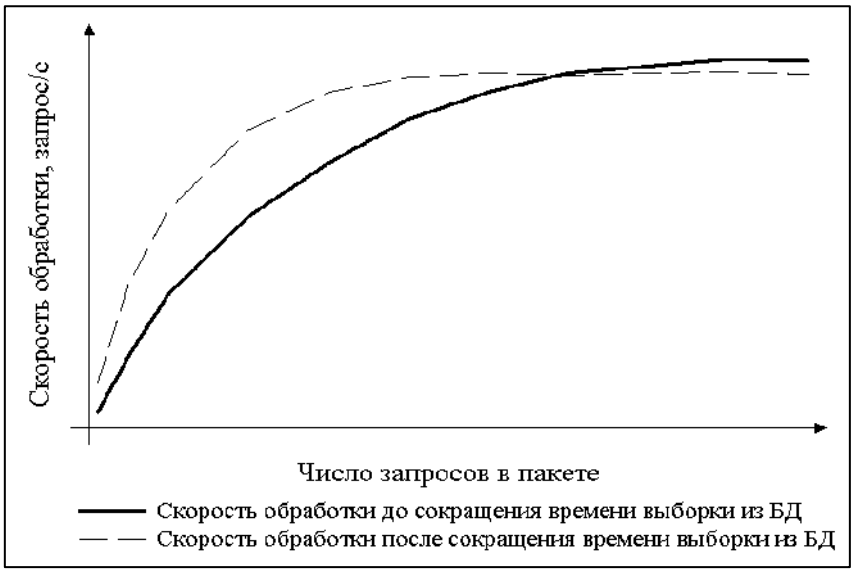


Рис. 1. Общий вид зависимости скорости обработки от размера пакета запросов

Данная работа представляет собой попытку создания детерминированной аналитической модели обработки пакета запросов в многопоточном сервере на сетевом узле с одним процессором. В отличие от стохастических моделей, построенных на основе теории очередей, эта модель должна учитывать особенности предметной области: высокую детерминированность длительности отдельных этапов обработки запросов и моментов событий, приводящих к переключению процессора с одной нити на другую. В отличие от других детерминированных моделей, построенных на основе теории расписаний [8-10], в данном случае целью является не поиск оптимального распределения работ по процессорам, но изучение зависимости основных характеристик программного сервера (время обработки пакета, предельная производительность) от характеристик запроса и размеров пакета. Основным понятием модели является модельная программная нить, отражающая основные черты поведения реальной программной нити: чередования состояний активной работы и состояний ожидания данных из внешних источников, периодов выполнения в процессоре и задержек (пауз), вызванных «конкуренцией» с другими нитями.

Описываемая модель основана на ряде упрощающих базисных предположениях о работе обрабатывающих нитей, отображенных на рис. 2. Самыми существенными из них являются 2-е и 4-е. Предположение о диспетчеризации нитей «без вытеснения» представляется оправданным по той причине, что длительности периодов активности в работе обрабатывающих нитей на сервере обычно не превышают несколько десятков миллисекунд. В этих условиях механизм «вытеснения», предназначенный для предотвращения монополизации процессора одной нитью, вряд ли был бы востребован. Предположение о стабильности времени выполнения операций, внешних по отношению к серверу приложений (например, поиска данных в сервере СУБД), вызвано необходимостью «очертить» границы модели только процессами, протекающими в сервере приложений, чтобы она не оказалась слишком громоздкой.

1. Модель обработки пакета запросов в многопоточном сервере

Так как целью моделирования является получение временных оценок, нас будут интересовать только временные характеристики обрабатывающих нитей. Будем считать, что каждая нить полностью описывается двумя наборами чисел

$$< W_1 , W_2 , \dots , W_m > u < t_1 , t_2 , \dots , t_{m+1} >,$$

где m – число периодов ожидания в работе нити, t_j $(j=1,2,\dots,m+1)$ – длительности периодов активности, W_j $(j=1,2,\dots, m)$ – длительности периодов ожидания, а поведение нити во времени описывается последовательностью: $t_1 , W_1 , t_2 , W_2 , \dots , t_m , W_m , t_{m+1}$, т.е. нить попеременно выполняет все периоды активности и периоды ожидания.

Соответственно, будем считать, что пакет запросов полностью описывается характеристиками обрабатывающей нити и числом запросов в пакете (n):

$$\{n, \quad < t_1 , W_1 , t_2 , W_2 , \dots , t_m , W_m , t_{m+1} >\}.$$

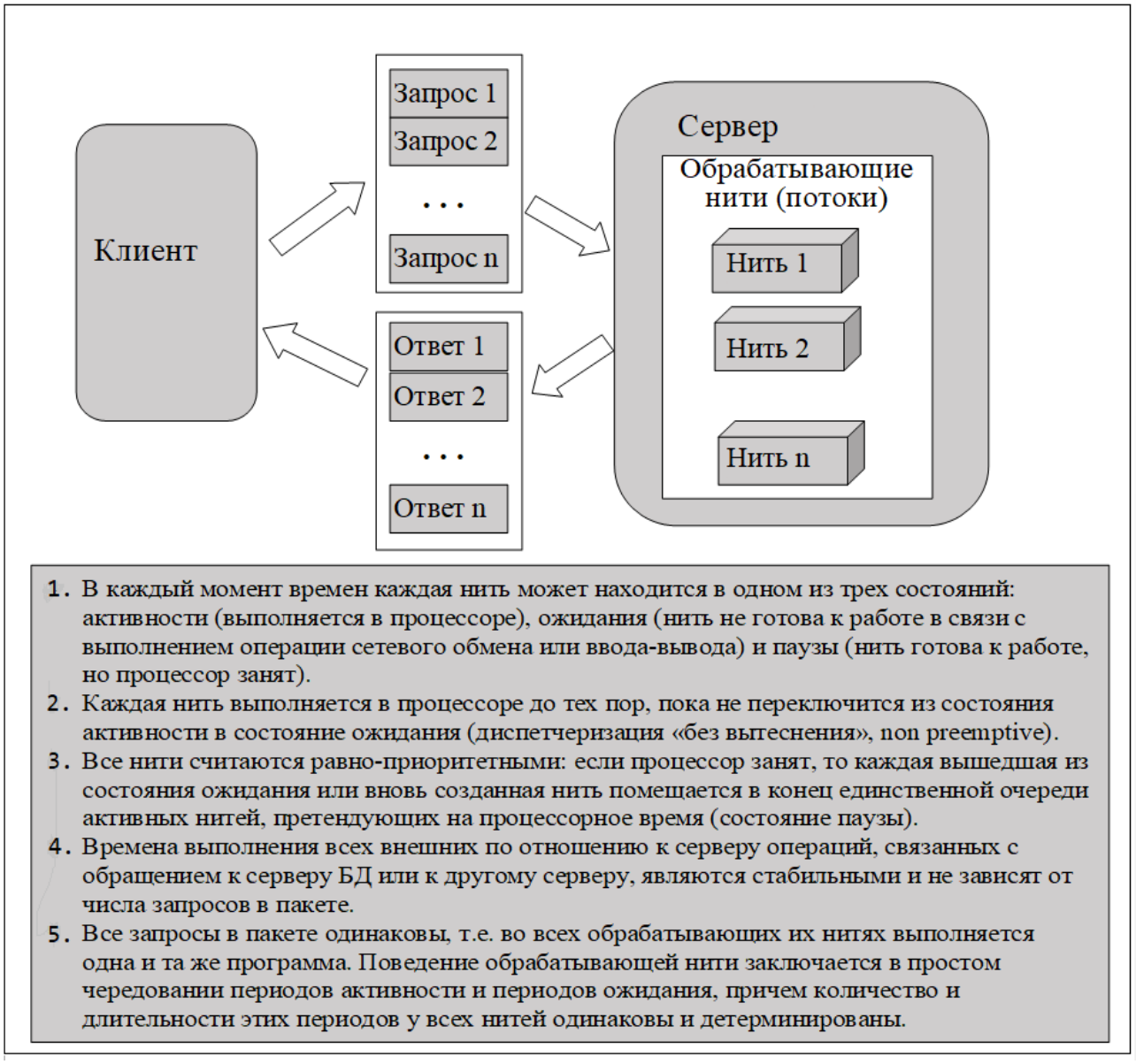
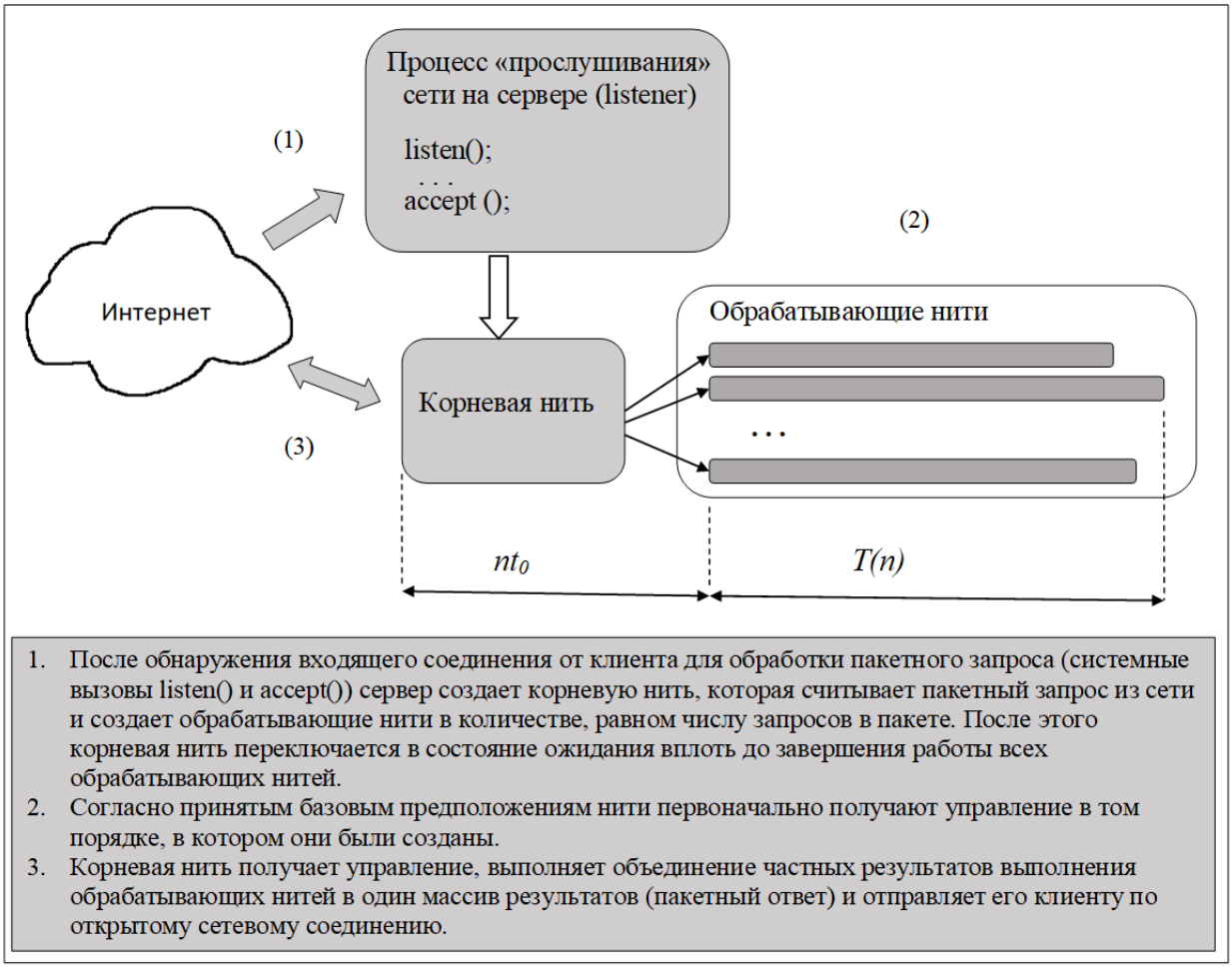


Рис. 2. Свойства модельных обрабатывающих нитей

Общая логика обработки пакета одновременных запросов в сервере и ее основные стадии отображены на рис. 3. Так как первая и третья стадии выполняется строго последовательно, то суммарное время их выполнения можно приближенно оценить, как nt_0, где n – число запросов в пакете, а t_0 – время потребное для создания одной обрабатывающей нити на стадии 1 и для обработки результата ее работы на стадии 3. Время обработки второй стадии $T(n)$ оценить сложнее, т.к. в процессе обработки возможны периоды ожидания и паузы в работе нитей, а также периоды простоя в работе процессора. Общее время обработки пакета запросов T_p представим в виде суммы:

$$T_p = nt_0 + T(n) \quad (1)$$

и попытаемся найти способы оценки $T(n)$, исходя из величин периодов активности t_i $(i=1,2, \ldots, m+1)$ и периодов ожидания W_i $(i=1,2, \ldots, m)$.



1. После обнаружения входящего соединения от клиента для обработки пакетного запроса (системные вызовы listen() и accept()) сервер создает корневую нить, которая считывает пакетный запрос из сети и создает обрабатывающие нити в количестве, равном числу запросов в пакете. После этого корневая нить переключается в состояние ожидания вплоть до завершения работы всех обрабатывающих нитей.
2. Согласно принятым базовым предположениям нити первоначально получают управление в том порядке, в котором они были созданы.
3. Корневая нить получает управление, выполняет объединение частных результатов выполнения обрабатывающих нитей в один массив результатов (пакетный ответ) и отправляет его клиенту по открытому сетевому соединению.

Рис. 3. Обработки пакета запросов в сервере

2. Основные свойства модели

При изучении свойств модели мы будем использовать двумерный числовой массив $[\tau_{i\,j}]$, в котором $\tau_{i\,j}$ представляет собой время начала выполнения *i*-й обрабатывающей нитью *j*-го интервала активности (здесь и далее будем считать, что нити перенумерованы в том порядке, в котором они были созданы). За начало отсчета времени принимается момент начала выполнения первой нитью первого интервала активности.

Утверждение 1. Пусть *n* – число запросов в пакете *(n>1)* и $1 \le i < j \le n$. Тогда $\tau_{i\,k} < \tau_{j\,k}$ при любом *k* $(1 \le k \le m+1)$.

Доказательство нетрудно провести с помощью индукции по значению *k*. Действительно, при *k=1* утверждение, очевидно, выполняется. Предположим, что $k > 1$ и $\tau_{i\,k-1} < \tau_{j\,k-1}$. Но нить *i* станет готова к выполнению периода активности *k* в момент времени (напомним, что мы рассматриваем диспетчеризацию «без вытеснения»), а нить *j* – в момент времени $\tau_{j\,k-1} + W_{k-1}$, т.е. нить i раньше встанет в очередь активных нитей, претендующих на процессорное время и раньше приступит к выполнению периода активности *k,* чем нить *j*. Доказательство закончено.

Как будет показано ниже, в рамках самой общей постановки модель позволила получить только нижнюю оценку T(n) сколько-нибудь приемлемой «компактности» (Утверждение 2). Для получения более точных оценок приходится накладывать дополнительные ограничения на характеристики пакета запросов и обрабатывающей нити (Утверждение 3 и Утверждение 4).

Утверждение 2. Если *m*>1 и *n*>1 и, то

$$T(n) \ge \max\left(max(nt_1,\ t_1 + W_1) + nt_{m+1} + \sum_{j=2}^{m} W_j\ ,\ nt_1 + \sum_{j=1}^{m} W_j + t_{m+1}\right) + \sum_{r=2}^{m} t_r \qquad (2)$$

(Здесь и далее, если нижний предел суммирования больше верхнего, сумма считается равной нулю.)

Пусть $\tau_{i\,j}$ – величина интервала времени от начала обработки до начала выполнения *i*-й обрабатывающей нитью *j*-го интервала активности *(i=1,2, …, n, j=1,…, m+1)*. Доказательство проведем

на основе системы рекуррентных соотношений, прямо вытекающих из свойств модели и Утверждения 1.

$$\begin{cases} \tau_{1\,1} = 0,\ \tau_{1\,2} = max(nt_1, t_1 + W_1) \\ \tau_{i\,1} = (i-1)t_1\,,\ \tau_{i\,j} \geq max\,(nt_1,\ \tau_{i\,j-1} + t_{j-1}\ + W_{j-1}\,)\ \ (j = 2,3,\dots,m) \\ \tau_{i\,m+1} = max(\tau_{i-1\,m+1} + t_{m+1}\,,\ \tau_{i\,m} + t_m\ + W_m\,)\ \ \ (i = 2,3,\dots,n) \end{cases} \quad (3)$$

Первая строка в системе рекуррентных соотношений описывает особые свойства первой обрабатывающей нити: она начинает выполнять первый интервал активности в момент времени 0, а второй интервал – в момент $t_1 + W_1$, но не раньше, чем все нити закончат первый интервал активности (nt_1), так как к моменту 0 все нити уже запущены и стоят в очереди к процессору.

Вторая строка описывает свойства любой нити: нить с номером *i* начнет выполнение первого интервала сразу после того, как все нити с меньшими номерами его закончат ($(i-1)t_1$), а каждый последующий интервал – не раньше, чем закончится выполнение предыдущего интервала активности и очередного интервала ожидания и не раньше, чем все нити закончат первый интервал активности (nt_1).

Наконец, третья строка связывает момент начала выполнения последнего интервала активности каждой нити *i* с моментом завершения последнего интервала активности предыдущей нитью *i*-1 и с моментом завершения нитью *i* предпоследнего интервала ожидания.

Сначала покажем, что при j=2, …, m выполняется соотношение

$$\tau_{i\,j} \geq \max\,(nt_1,\ it_1 + W_1) + \sum_{r=2}^{j-1} W_r + \sum_{r=2}^{j-1} t_r. \quad (4)$$

Доказательство несложно провести с помощью индукции по *j*. При j=2 справедливость соотношения (4) вытекает непосредственно из (3). Предположим, что (4) выполняется при *j*=*k* (1<*k*<*m*) и докажем, что оно выполняется при *j*=*k*+1. В самом деле, согласно (3) $\tau_{i\,k+1} \geq\ \tau_{i\,k} + t_k\ + W_k =$
$\max(nt_1,\ it_1 + W_1\,) + \sum_{r=2}^{k-1} W_r + \sum_{r=2}^{k-1} t_r + t_k\ + W_k\ =\ \max(nt_1,\ it_1 +\ W_1\,) + \sum_{r=2}^{k} W_r + \sum_{r=2}^{k} t_r.$

Докажем теперь, что

$$\tau_{i\,m+1} \geq \max(\max(nt_1,\ t_1 + W_1) + (i-1)t_{m+1} + \sum_{r=2}^{m} W_r + \sum_{r=2}^{m} t_r\,,\ \max\,(nt_1,\ it_1 +\ W_1\,) + \sum_{r=2}^{m} W_r + \sum_{r=2}^{m} t_r). \quad (5)$$

Доказательство проведем с помощью индукции по i. Легко видеть, что при *i*=1 (5) сокращается до $\tau_{i\,m+1} \geq \max(nt_1,\ t_1 + W_1) + \sum_{r=2}^{m} W_r + \sum_{r=2}^{m} t_r$. Справедливость этого соотношения вытекает из (4).

Предположим теперь, что (5) выполняется при *i = k (1 ≤ k < n)*. Тогда при *i*=*k+1* с учетом (3) и (4) имеем:

$$\tau_{k+1\,m+1} \geq max(\tau_{k\,m+1} + t_{m+1}\,,\ \tau_{k+1\,m} + t_m\ + W_m\ ) \geq$$

$$max(max(max(nt_1,\ t_1 + W_1) + (k-1)t_{m+1} + \sum_{r=2}^{m} W_r + \sum_{r=2}^{m} t_r\,, max(nt_1, kt_1\ + W_1\,) + \sum_{r=2}^{m} W_r + \sum_{r=2}^{m} t_r\,) + t_{m+1}\,,\ max(nt_1, (k+1)t_1\ + W_1) + t_m\ + \sum_{r=2}^{m} W_r + \sum_{r=2}^{m} t_r) \geq$$

$$max(max(max(nt_1,\ t_1 + W_1) + kt_{m+1} + \sum_{r=2}^{m} W_r + \sum_{r=2}^{m} t_r\,, max(nt_1, kt_1\ + W_1\,) + \sum_{r=2}^{m} W_r\ + \sum_{r=2}^{m} t_r),\ max(nt_1, (k+1)t_1\ + W_1) + \sum_{r=2}^{m} W_r + \sum_{r=2}^{m} t_r) =$$

$$max(max(nt_1,\ t_1 + W_1) + kt_{m+1} + \sum_{r=2}^{m} W_r + \sum_{r=2}^{m} t_r,\ \ max(nt_1, kt_1\ + W_1\,) + \sum_{r=2}^{m} W_r + \sum_{r=2}^{m} t_r\,,\ max(nt_1, (k+1)t_1\ + W_1) + t_m\ + \sum_{r=2}^{m} W_r + \sum_{r=2}^{m} t_r)\,.$$

Учитывая, что $\max(nt_1, kt_1\ + W_1\,) + \sum_{r=2}^{m} W_r + \sum_{r=2}^{m} t_r\ \leq\ max(nt_1, (k+1)t_1\ + W_1) + \sum_{r=2}^{m} W_r + \sum_{r=2}^{m} t_r$, имеем:
$\tau_{k+1\,m+1}\ \geq\ max(max(nt_1,\ t_1 + W_1) + kt_{m+1} + \sum_{r=2}^{m} W_r + \sum_{r=2}^{m} t_r\,,\ max(nt_1, (k+1)t_1\ + W_1) + \sum_{r=2}^{m} W_r + \sum_{r=2}^{m} t_r)$
Т.е. в этом случае (7) выполняется при *i*=*k+1*.

Отсюда: $T(n) \geq \tau_{n\,\mathrm{m+1}} +\ t_{\mathrm{m+1}} = \max(\max(nt_1,\ t_1 + W_1) + \sum_{r=2}^{m} W_r + \sum_{r=2}^{m} t_r + nt_{\mathrm{m+1}}\ ,\ \max\,(nt_1,\ nt_1 + W_1) + \sum_{r=2}^{m} W_r + \sum_{r=2}^{m} t_r\ +\ t_{m+1}) = \max\,(\max(nt_1,\ t_1 + W_1) + \sum_{r=2}^{m} W_r + \sum_{r=2}^{m} t_r + nt_{\mathrm{m+1}}\ ,\ nt_1 +\ \sum_{r=1}^{m} W_r + \sum_{r=2}^{m} t_r\ +\ t_{m+1})$. Доказательство закончено.

Следующее утверждение говорит о возможности точной оценки $T(n)$ в случае, если обрабатывающая нить имеет один период ожидания и два периода активности.

Утверждение 3. Если *m*=1, то

$$T(n) = \max(\max(nt_1,\ t_1 + W_1) + nt_2\ ,\ nt_1 +\ W_1 +\ t_2) \quad (6)$$

Пусть $\tau_{i\,j}$ по-прежнему – величина интервала времени от начала обработки до начала выполнения *i*-й обрабатывающей нитью *j*-го интервала активности (i=1,2,…, n, j=1,…, m+1). Доказательство проведем на основе системы рекуррентных соотношений, прямо вытекающих из свойств модели и Утверждения 1.

$$\begin{cases} \tau_{1\,1} = 0,\ \tau_{1\,2} = max(nt_1,\ t_1 + W_1) \\ \tau_{i\,1} = \tau_{i-1\,1} + t_1,\ \ \tau_{i\,2} = max(\tau_{i-1\,2} + t_2\,,\ max(nt_1,\ \tau_{i\,1} + t_1 + W_1)) \quad (i = 2, \dots, n) \end{cases} \tag{7}$$

Первая строка в системе рекуррентных соотношений описывает особые свойства первой обрабатывающей нити: она начинает выполнять первый интервал активности в момент времени 0, а второй интервал – в момент $t_1 + W_1$, но не раньше, чем все нити закончат первый интервал активности (nt_1), так как к моменту 0 все нити уже запущены и стоят в очереди к процессору.

Вторая строка связывает момент начала выполнения первого интервала активности каждой нити *i* с моментом завершения первого интервала активности предыдущей нитью *i*-1, а также связывает момент начала выполнения второго интервала активности каждой нити *i* с моментом завершения второго интервала активности предыдущей нитью *i*-1 и с моментом завершения нитью *i* единственного интервала ожидания.

Отметим, что из $\tau_{1\,1} = 0$ и $\tau_{i\,1} = \tau_{i-1\,1} + t_1$ прямо следует, что $\tau_{i\,1} = (i-1)t_1\ \ (i = 1, \dots, n)$.

Докажем теперь, что

$$\tau_{i\,2} = \max(\max(nt_1,\ t_1 + W_1) + (i-1)\,t_2\,,\ \max\,(nt_1,\ it_1\ + W_1)\,) \tag{8}$$

Доказательство проведем с помощью индукции по i. Легко видеть, что при *i*=1 (7) сокращается до $\tau_{i\,2} = \max(nt_1,\ t_1 + W_1)$, что соответствует (7).

Предположим теперь, что (8) выполняется при *i = k (1 ≤ k < n)*. Тогда при *i=k+1* имеем

$$\tau_{k+1\,2} = max(\tau_{k\,2} + t_2\,,\ \ max(nt_1,\ (k+1)t_1\ + W_1)) =$$

$$max(max(max(nt_1,\ t_1 + W_1) + (k-1)t_2\ , max\,(nt_1, kt_1\ + W_1)\,) + t_2\,, max(nt_1, (k+1)t_1\ + W_1) =$$

$$max(max\big(max(nt_1,\ t_1 + W_1) + kt_2\ \ , max(nt_1, kt_1 + W_1) + t_2\ \big),\ max\,(nt_1, (k+1)t_1\ + W_1)) =$$

$$max\big(max(nt_1,\ t_1 + W_1) + kt_2\ \ ,\ max(nt_1,\ kt_1 + W_1) + t_2\,,\ max(nt_1,\ (k+1)t_1\ + W_1)\big). \tag{9}$$

Предположим, что $t_2\ \geq\ t_1$. Тогда $max(nt_1,\ t_1 + W_1) + kt_2\ \geq\ max(nt_1, kt_1 + W_1) +\ t_2$ и (9) принимает простую форму:

$$max(max(nt_1,\ t_1 + W_1) + kt_2\,)\,,\ max\,(\,nt_1,\ (k+1)t_1\ + W_1)\,). \tag{10}$$

Предположим теперь, что $t_2\ <\ t_1$ и $\ t_1 + W_1 \leq\ nt_1 <\ \ kt_1 + W_1$. Легко видеть, что в этом случае $max(nt_1,\ kt_1 + W_1\,) + t_2\ \leq\ max(nt_1,\ (k+1)t_1\ + W_1\,)$ и (9) также принимает форму (10). Если же $nt_1 <\ t_1 + W_1$ или $nt_1\ \geq\ kt_1 + W_1$, то $max(nt_1,\ t_1 + W_1) + kt_2\ \geq\ max(nt_1, kt_1 + W_1) + t_2$, а (9) и в этом случае принимает форму (10). Таким образом, (9) тождественно (10). На этом доказательство равенства (8) закончено. (В доказательстве использовано очевидное правило: если выполняется: а $\geq b$ или $b\ \leq c,$ то $\max(a, b, c) = \max\,(a, c)$. В данном случае роль *a, b* и *c* играют соответственно три аргумента внешней функции *max* в (9). Вышеприведенное рассуждение доказывает, что значение второго аргумент внешней функции *max* никогда не превосходит значений и первого и второго аргументов сразу. Другими словами, второй аргумент является попросту «лишним» и (9) тождественно (10)).

Поэтому имеем: $T(n) = \tau_{n\,2} +\ t_2 = \max\,(\max(nt_1,\ t_1 + W_1) + nt_2\ ,\ \max\,(nt_1,\ nt_1 +\ W_1 +\ t_2)\,) =$ $\max(\max(nt_1,\ t_1 + W_1) + nt_2\ ,\ nt_1 +\ W_1 +\ t_2\,)$. Доказательство закончено.

Следствие 1. Если имеется единственный период ожидания и

$$n \geq max\,(\tfrac{W_1+t_1}{t_1}, \tfrac{W_1+t_2}{t_2})\,), \tag{11}$$

то $T(n) = nt_1 + nt_2$ и $T_p = nt_0 + T(n) = nt_0 + nt_1 + nt_2$.

Доказательство. Если выполняется (11), то, в соответствии с (6)
$T(n) = \max(max(nt_1,\ t_1 + W_1) + nt_2\ ,\ nt_1 +\ W_1 +\ t_2) = \max(nt_1 + nt_2\,,\ nt_1 +\ W_1 +\ t_2\,) =\ \ nt_1 + nt_2$.

Отсюда $\ T_p = nt_0 + T(n) = nt_0 + nt_1 + nt_2$.

На рис. 4 приведен общий вид зависимости скорости обработки n/T_p = *n/(nt₀+T(n))* от размера пакета *n*, и показано, каким образом меняется вид кривой при сокращении периода ожидания. Если сравнить рис. 4 с рис. 1, то можно заметить определенное сходство в поведении кривых.

Как видно из рисунка, обе кривые в целом «ведут себя» примерно одинаково. При малых значениях n наблюдается быстрый рост скорости (положительный эффект от пакетной обработки!), потом кривая становится все более «пологой» и наконец достигает предельного значения, общего для обеих кривых. Согласно Следствию 1 Утверждения 3 это значение равно $\frac{n}{nt_0+\mathrm{n}t_1+\mathrm{n}\,t_2} = \frac{1}{t_0+t_1+t_2}$ запросов в единицу времени. Единственная разница между кривыми заключается в том, что вторая (кривая (отображенная пунктирной линией) раньше достигает предельного значения, чем первая. Аналогичные черты поведения кривых наблюдаются и на рис. 1.

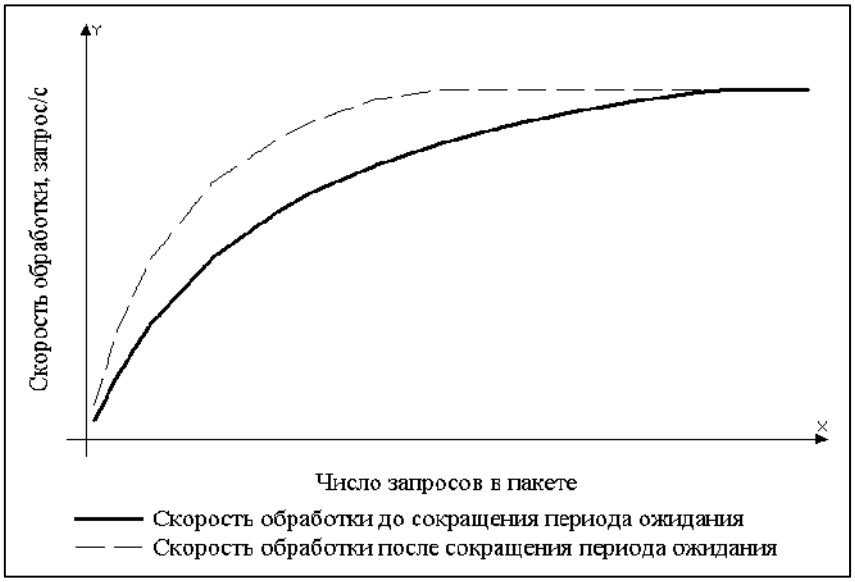


Рис. 4. Общий вид зависимости модельной скорости обработки от размера пакета

В рассмотренном ниже Утверждении 4 предполагается, что длительности всех интервалов активности, кроме первого и последнего, равны нулю. Это предположение максимально приближает ситуацию, рассмотренную в Утверждении 4, к ситуации с единственным периодом ожидания, рассмотренной в Утверждении 3.

Утверждение 4. Если *m*>1 и *n*>1 и $t_j = 0$ при *j*=2,…, m, то

$$T(n) \geq \max\left(\max(nt_1,\ t_1 + W_1) + nt_{m+1} + \sum_{j=2}^{m} W_j\ ,\ nt_1 + \sum_{j=1}^{m} W_j + t_{m+1}\right) \tag{12}$$

и

$$T(n) \leq \max\left(\max(nt_1,\ t_1 + W_1) + nt_{m+1} + \sum_{j=2}^{m} W_j\ ,\ nt_1 + \sum_{j=1}^{m} W_j + nt_{m+1}\right). \tag{13}$$

Пусть $\tau_{i\,j}$ по-прежнему – величина интервала времени от начала обработки до начала выполнения *i*-й обрабатывающей нитью *j*-го интервала активности *(i=1,2, …, n, j=1,…, m=1)*. Доказательство проведем на основе системы рекуррентных соотношений, прямо вытекающих из свойств модели и Утверждения 1.

$$\begin{cases} \tau_{1\,1} = 0, \tau_{1\,2} = max(nt_1, t_1 + W_1) \\ \tau_{i\,1} = (i-1)t_1\ ,\ \tau_{i\,j} \geq max\ (nt_1,\ \tau_{i\,j-1} + t_{j-1}\ + W_{j-1}\ )\ \ (j = 2,3,\dots,m) \\ \tau_{i\,m+1} = max(\tau_{i-1\,m+1} + t_{m+1}\ ,\ \tau_{i\,m} + t_m\ + W_m\ )\quad (i = 2,3,\dots,n) \end{cases} \tag{14}$$

Первая строка в системе рекуррентных соотношений описывает особые свойства первой обрабатывающей нити: она начинает выполнять первый интервал активности в момент времени 0, а второй интервал – в момент $t_1 + W_1$, но не раньше, чем все нити закончат первый интервал активности (nt_1), так как к моменту 0 все нити уже запущены и стоят в очереди к процессору.

Вторая строка описывает свойства любой нити: нить с номером *i* начнет выполнение первого интервала сразу после того, как все нити с меньшими номерами его закончат ($(i-1)t_1$), а каждый последующий интервал – не раньше, чем закончится выполнение предыдущего интервала активности

и очередного интервала ожидания и не раньше, чем все нити закончат первый интервал активности (nt_1).

Наконец, третья строка связывает момент начала выполнения последнего интервала активности каждой нити *i* с моментом завершения последнего интервала активности предыдущей нитью *i*-1 и с моментом завершения нитью *i* предпоследнего интервала ожидания.

Сначала покажем, что при j=2, …, m выполняется соотношение

$$\tau_{i\,j} \geq max(nt_1,\ it_1 + W_1) + \sum_{r=2}^{j-1} W_r). \quad (15)$$

Доказательство несложно провести с помощью индукции по *j*. При j=2 справедливость соотношения (15) вытекает непосредственно из (5). Предположим, что (15) выполняется при *j=k* (1<*k*<*m*) и докажем, что оно выполняется при *j=k*+1. В самом деле, согласно (12) $\tau_{i\,k+1} \geq \tau_{i\,k} + t_k + W_k$ =max($nt_1,\ it_1 + W_1$) + $\sum_{r=1}^{k-1} W_r + t_k + W_k$ = max($nt_1,\ it_1 + W_1$) + $\sum_{r=1}^{k} W_r$ так как $t_k = 0$ при k=2, …, m.

Докажем теперь, что

$$\tau_{i\,m+1} \geq max(max(nt_1,\ t_1 + W_1) + (i-1)t_{m+1} + \sum_{r=2}^{m} W_r\ ,\ max(nt_1, it_1 + W_1) + \sum_{r=2}^{m} W_r\ ). \quad (16)$$

Доказательство проведем с помощью индукции по i. Легко видеть, что при *i*=1 (7) сокращается до $\tau_{i\,m+1} \geq \max(nt_1,\ t_1 + W_1) + \sum_{j=2}^{m} W_j$. Справедливость этого соотношения вытекает из (6).

Предположим теперь, что (16) выполняется при *i = k (1 ≤ k < n)*. Тогда при *i=k+1* с учетом (14), (15) и того, что $t_j = 0$ при *j*=2, …, m, имеем

$$\tau_{k+1\,m+1} \geq max(\tau_{k\,m+1} + t_{m+1},\ \tau_{k+1\,m} + t_m + W_m\ ) \geq$$

$$max(max(max(nt_1,\ t_1 + W_1) + (k-1)t_{m+1} + \sum_{r=2}^{m} W_r\ , max(nt_1, kt_1 + W_1\ ) + \sum_{r=2}^{m} W_r\ ) + t_{m+1}\ ,\ max(nt_1, (k+1)t_1 + W_1) + t_m + \sum_{r=2}^{m} W_r) \geq$$

$$max(max(max(nt_1,\ t_1 + W_1) + kt_{m+1} + \sum_{r=2}^{m} W_r\ , max(nt_1, kt_1 + W_1\ ) + \sum_{r=2}^{m} W_r\ ),\ max(nt_1, (k+1)t_1 + W_1) + \sum_{r=2}^{m} W_r) =$$

$$max(max(nt_1,\ t_1 + W_1) + kt_{m+1} + \sum_{r=2}^{m} W_r,\ \ max(nt_1, kt_1 + W_1\ ) + \sum_{r=2}^{m} W_r\ ,\ max(nt_1, (k+1)t_1 + W_1) + t_m + \sum_{r=2}^{m} W_r)\ .$$

Учитывая, что $\max(nt_1, kt_1 + W_1\ ) + \sum_{r=2}^{m} W_r \leq max(nt_1, (k+1)t_1 + W_1) + \sum_{r=2}^{m} W_r$, имеем: $\tau_{k+1\,m+1} \geq max(max(nt_1,\ t_1 + W_1) + kt_{m+1} + \sum_{r=2}^{m} W_r\ ,\ max(nt_1, (k+1)t_1 + W_1) + \sum_{r=2}^{m} W_r)$ Т.е. в этом случае (16) выполняется при *i=k+1*.

Поэтому имеем: $T(n) \geq \tau_{n\,\mathrm{m+1}} + t_{\mathrm{m+1}}$ = max(max($nt_1,\ t_1 + W_1$) + $\sum_{r=2}^{m} W_r + nt_{\mathrm{m+1}}$, max (nt_1, $nt_1 + W_1$) + $\sum_{r=2}^{m} W_r + t_{m+1}$) = max (max($nt_1,\ t_1 + W_1$) + $\sum_{r=2}^{m} W_r + nt_{\mathrm{m+1}}$, $nt_1 + \sum_{r=1}^{m} W_r + t_{m+1}$) . Т.е. справедливость (12) доказана.

Докажем теперь справедливость (13). Перепишем 2-ю и 3-ю строки системы рекуррентных соотношений (14) в следующей форме:

$$\begin{cases} \tau_{i\,1} = (i-1)t_1\ ,\ \tau_{i\,j} = max\ (nt_1,\ \tau_{i\,j-1} + \delta_{i\,j-1} + t_{j-1} + W_{J-1}\ )\ \ (j = 2,3,\dots,m) \\ \tau_{i\,m+1} = max(\tau_{i-1\,m+1} + t_{m+1},\ \tau_{i\,m} + t_m + W_m\ ), \end{cases},$$

где $\delta_{i\,j} \geq 0$ – величина вынужденной паузы в работе обрабатывающей нити *i* перед началом выполнения *j*-го интервала активности вследствие занятости процессора (отметим, что попытка выполнить сколь угодно малый интервал активности может привести к длительной паузе, если процессор занят длительной работой).

В результате формула (16) примет следующий вид:

$$\tau_{i\,m+1} = max(max(nt_1,\ t_1 + W_1) + (i-1)t_{m+1} + \sum_{r=2}^{m} W_r + \sum_{r=2}^{m} t_r\ ,\ max\ (nt_1,\ it_1 + \sum_{r=1}^{m} W_r + \sum_{r=2}^{m} t_r + \sum_{r=2}^{m} \delta_{i\,r})),$$

(Доказательство нетрудно провести аналогично доказательству формулы (16) с помощью индукции по *i*.)

Отсюда:

$$\tau_{n\,m+1} = max(max(nt_1,\ t_1 + W_1) + (n-1)t_{m+1} + \sum_{r=2}^{m} W_r + \sum_{r=2}^{m} t_r\ ,\ nt_1 + \sum_{r=1}^{m} W_r + \sum_{r=2}^{m} t_r + \sum_{r=2}^{m} \delta_{n\,r})$$

Если $t_j = 0$ при j=2, …, m, то $\sum_{r=2}^{m} \delta_{n\,r}$ не может превысить $(n-1)t_{m+1}$, т.к. к моменту выполнению последней нитью интервалов активности с номерами от 2 до m все нити уже завершили выполнение первого интервала активности (см. Утверждение 1), а $\sum_{r=2}^{m} t_r$=0.

В результате имеем:

$$T(n) = \tau_{n\,m+1} + t_{m+1} = max(max(nt_1,\ t_1 + W_1) + \sum_{r=2}^{m} W_r + \sum_{r=2}^{m} t_r + (n-1)t_{m+1}\ ,\ \ nt_1 + \sum_{r=1}^{m} W_r + \sum_{r=2}^{m} t_r + \sum_{r=2}^{m} \delta_{n\,r}) + t_{m+1} \le max(max(nt_1,\ t_1 + W_1) + \sum_{r=2}^{m} W_r + (n-1)t_{m+1}\ ,\ nt_1 + \sum_{r=1}^{m} W_r + (n-1)t_{m+1}) + t_{m+1} = max\ (max(nt_1,\ t_1 + W_1) + \sum_{r=2}^{m} W_r + nt_{m+1}\ ,\ nt_1 + \sum_{r=1}^{m} W_r + nt_{m+1})\ .$$

Доказательство закончено.

Утверждение 5. Если m>1, n>1 и $t_j = 0$ при j=2,…, m, а $W_m \ge W_1$, то

$$T(n) = max\big(max(nt_1,\ t_1 + W_1) + nt_{m+1} + \sum_{j=2}^{m} W_j\ ,\ nt_1 + \sum_{j=1}^{m} W_j + t_{m+1}\big)\ . \qquad (17)$$

Если длительности всех интервалов активности, кроме первого и последнего, равны нулю, то $\tau_{1\,m+1}$ (время начала выполнения последнего интервала активности первой нитью) можно выразить формулой $\tau_{1\,m+1} = max(nt_1,\ t_1 + W_1) + \sum_{j=2}^{m} W_j$. Сопоставим $\tau_{1\,m+1}$ с $\tau_{n\,m}$ (временем начала выполнения предпоследнего интервала активности последней нитью). Если $\tau_{1\,m+1} > \tau_{n\,m}$, то все интервалы активности со второго по предпоследний будут выполнены последней нитью без дополнительных задержек, связанных с ожиданием освобождения процессора, так как к моменту их выполнения все нити уже завершили выполнение первого интервала активности и ни одна нить еще не начала выполнения последнего интервала активности (а только эти интервалы ненулевые). Поэтому, в этом случае $\tau_{n\,m} = nt_1 + \sum_{j=1}^{m-1} W_j$. Но при $W_m > W_1$имеем $\tau_{1\,m+1} = max(nt_1,\ t_1 + W_1) + \sum_{j=2}^{m} W_j \ge nt_1 + \sum_{j=2}^{m} W_j > nt_1 + \sum_{j=1}^{m-1} W_j = \tau_{n\,m}$. Другими словами, в этом случае в формуле (12) можно заменить знак ≥ на знак равенства и получить точную оценку для $T(n)$ (17)..

Утверждение 6. Если *n>1, m>1* и W_k*>0 (k=1,2,…,m)*, то время выполнения пакета запросов, удовлетворяющего условию Следствия 1 из Утверждения 2, (T_1) больше или равно времени выполнения равного по размеру пакета простых запросов (T_2) с двумя периодами активности, равными t_1 (начальный) и t_{m+1} (конечный) и единственным периодом ожидания, равным $\sum_{k=1}^{m} W_k$. При этом при достаточно больших n имеет место строгое неравенство: $T_1 > T_2$.

Доказательство. Как следует из Утверждения 2,

$$T_1 \ge max\ (max(nt_1,\ t_1 + W_1) + nt_{m+1} + \sum_{j=2}^{m} W_j\ ,\ nt_1 + \sum_{j=1}^{m} W_j + t_{m+1})\ .$$

$$T_2 = max\ (max\big(nt_1,\ t_1 + \sum_{j=1}^{m} W_j\big) + nt_{m+1}\ ,\ nt_1 + \sum_{j=1}^{m} W_j + t_{m+1})\ .$$

Разобьём множество возможных размеров пакета (n) на 3 непересекающихся интервала:

$1 \le n \le W_1/t_1 + 1$, $\quad W_1/t_1 + 1 < n \le (\sum_{k=1}^{m} W_k)/t_1 + 1$ и $n > (\sum_{k=1}^{m} W_k)/t_1 + 1$.

Для каждого интервала значений n выражения для T_1 и T_2 могут быть упрощены путем несложных преобразований. В таблице. 1 приведены результаты таких преобразований для каждого интервала значений n и вытекающее из них соотношение T_1 и T_2.

Таблица 1. Соотношение времен обработки

Условие	**Значения *T1* и *T2***	**Соотношение T_1 и T_2**
$1 \le n \le W_1/t_1 + 1$	$T_1 = t_1 + \sum_{k=1}^{m} W_k + t_{m+1} + (n-1)\ max(t_1,\ t_{m+1})$ $T_2 = t_1 + \sum_{k=1}^{m} W_k + t_{m+1} + (n-1)\ max(t_1,\ t_{m+1})$	$T_1 \ge T_2$
$W_1/t_1 + 1 < n \le (\sum_{k=1}^{m} W_k)/t_1 + 1$	$T_1 = nt_1 + \sum_{k=1}^{m} W_k + max(nt_{m+1} - W_1,\ t_{m+1})$ $T_2 = nt_1 + \sum_{k=1}^{m} W_k + max(nt_{m+1} - (n-1)t_1,\ t_{m+1})$	Если $t_{m+1} > t_1$, то $T_1 > T_2$ иначе $T_1 \ge T_2$
$> (\sum_{k=1}^{m} W_k)/t_1 + 1$	$T1 = nt_1 + \sum_{k=1}^{m} W_k + max(nt_{m+1} - W1,\ t_{m+1})$ $T2 = nt_1 + \sum_{k=1}^{m} W_k + max(nt_{m+1} - \sum_{k=1}^{m} W_k\ ,\ t_{m+1})$	Если $(n-1)t_{m+1} > W_1$, то $T_1 > T_2$, иначе $T_1 \ge T_2$

Легко видеть, что, при $n > max\ ((\sum_{k=1}^{m} W_k + t_1)/t_1,\ (\sum_{k=1}^{m} W_k + t_{m+1})/t_{m+1})$ значения всех функций *max* в выражениях для T_1, *и* T_2 равны их первому аргументу. А это означает, что $T_1 = nt_1 + \sum_{k=2}^{m} W_k + nt_{m+1}$ и $T_2 = nt_1 + nt_{m+1}$. То есть, $T_1 > T_2$.

Утверждение означает, что при обработке пакетов запросов один большой период ожидания предпочтительнее, чем несколько малых, равных ему в сумме, даже если последние разделены пренебрежимо малыми периодами активности.

3. Заключение

Обнаружение определенного сходства в поведения кривых, отражающих зависимость скорости обработки от размера пакета запросов, полученных экспериментально (рис. 1) и аналитически (рис. 4) явилось для автора одним из самых важных этапов исследования. Оно подтвердило, что несмотря на целый ряд упрощающих предположений, допущенных при построении модели (см. раздел 1), полного «разрыва связи» модели с реальностью все-таки не произошло. Другими словами, описанная аналитическая модель действительно позволяет получить хотя бы приближенные оценки быстродействия программного сервера при обработке пакетов запросов на основании характеристик одиночного запроса и размеров пакета.

Разработчикам распределенных систем хорошо знают об эффективности использования т.н. «хранимых процедур» при создании сервисов, обращающихся к СУБД [11-13]. Эта эффективность обусловлена чисто техническими факторами: информационные запросы к базам данных (SQL-запросы), размещенные в хранимых процедурах, как правило выполняются несколько быстрее, чем запросы, поступившие извне. Утверждение 6 также приводит к важному практическому выводу о выигрыше, связанном с использованием хранимых процедур, но в данном случае речь идет о концептуальном выигрыше, вытекающим из описанной модели пакетной обработки. В самом деле, размещение нескольких SQL-запросов в хранимой процедуре позволяет заменить множество «малых» периодов ожидания, связанных с выполнением отдельных SQL-запросов, одним «большим» периодом ожидания, связанным с выполнением всей процедуры, в каждой обрабатывающей нити. Как видно из Утверждения 6, положительный эффект имеет нетривиальный характер: например, время обработке одиночного запроса не изменится от такой замены. Но при обработке пакетных запросов возникает разница и возникает концептуальный выигрыш.

Литература

1. *Muter I.* Exact algorithms to minimize makespan on single and parallel batch processing machines // European Journal of Operational Research. – 2020. – Vol. 285, № 2. – P. 470–483.
2. *Ding S., Attenberg J., Baeza-Yates R., Suel T.* Batch query processing for web search engines, // Proceedings of the fourth ACM international conference on Web search and data mining. – NY: ACM, 2011. – P. 137–146.
3. *Tannenbaum A., Bos H.* Modern Operating Systems. – N.J.: Pearson Education, 2015. – 1136 p.
4. *Келли-Бутл С.* Введение в Unix. – М.: ЛОРИ, 1995. – 596 с.
5. *Волосатова Т.М., Киселев И.А., Князева С.В.* Многопоточная обработка в POSIX стандарте // Восточно-Европейский научный журнал. – 2021. – № 12–3. – С. 37–39.
6. *Larina T.B., Filipchenko A.S.* Experiments with multi-threaded processing in RPA Robin // Information Innovative Technologies: International Scientific-Practical Conference, Prague. – Moscow: Association of graduates and employees of AFEA named after prof. Zhukovsky, 2022. – P. 245–252.
7. *Ткаченко В.И.* Перспективы развития операционных систем // Научные достижения и инновации: вопросы теории и практики: Материалы XIV Всероссийской научно-практической конференции, Ростов-на-Дону, 15 сентября 2022 года. – Ростов-на-Дону: Параграф, 2022. – С. 46–49.
8. *Коффман Э.Г.* Теория расписаний и вычислительные машины. – М.: Наука, 1984. – 336 с.
9. *Chernykh K.A., Servakh V.V.* Combinatorial structure of optimal solutions to the problem of a single machine with preemption // 15th International Asian School-Seminar Optimization Problems of Complex Systems, OPCS 2019. – Novosibirsk, 2019. – P. 21–26.
10. *Romanova A.A., Servakh V.V.* Complexity of cyclic job shop scheduling problem for identical jobs with no-wait constraints // Journal of Applied and Industrial Mathematics. – 2019. – Vol. 13, № 4. – P. 706–716.
11. *Прибыл Б., Фейерштейн С.* Oracle PL/SQL для профессионалов. – СПб: Питер, 2015. – 1024 с.
12. *Feuerstein S.* Oracle PL/SQL Programming, O'Reilly, 2014. – 1392 p.
13. *Ferrari L., Pirozzi E.* Learn PostgreSQL. – Birmingham: Packt, 2020. – 650 p.