

## МОДЕЛИРОВАНИЕ ТРАФИКОВ И ОЦЕНКА СКОРОСТИ РАСПРЕДЕЛЁННОГО ДОСТУПА В СИСТЕМАХ ОБЛАЧНЫХ ВЫЧИСЛЕНИЙ С ОБЩИМ РЕСУРСОМ НА ПРИМЕРЕ ORACLE RAC

Проводится анализ применимости математической модели функционирования распределённой общей памяти в облачных системах на примере реализации Shared Everything подхода в Oracle RAC. Проектируется математическая модель тестового трафика для работы с моделью, а также определена целевая функция для анализа эффективности алгоритмов оптимизации доступа к глобальной странице памяти в облачной системе.

The paper analyzes mathematical model of functioning distributed shared memory in a cloud computing systems by the example of the Oracle RAC. There is designed mathematical model of test traffic for using inside proposed model. Also defined target function for analysis of global memory page access effectiveness in a cloud computing systems.

### Постановка задачи

В работе [1] была предложена математическая модель функционирования распределённой общей памяти в облачных системах. При этом целевой функцией оценки оптимизации принято считать среднее время доступа к странице памяти (блоку). Время доступа, как было показано в работе [1], зависит от трех категорий параметров: базовых характеристиками системы, т. е. тех, которые зависят только от архитектуры системы и физических возможностей, базовых характеристик трафика т. е. тех, которые зависят только от характера трафика, и производных параметров, таких, которые обусловлены взаимодействием трафика с системой. Например, скорость передачи данных между нодами, количество нод или размер кэша на каждой ноде – это базовые характеристики системы, интенсивность запросов, соотношение

запросов чтения и записи, вероятность конфликта – базовые характеристиками трафика, а вероятность попадания в кэш или среднее время блокирования – производные параметры. Задачей этой работы будет, проанализировав существующие в современных системах трафики, выработать методику описания базовых характеристик трафика и методику вычисления производных параметров, после чего показать пригодность модели для анализа эффективности облачных систем с распределённой памятью.

### Резюме математической модели

Приведём полученную в работе [1] формулу для оценки среднего времени доступа к блоку и распишем значения параметров, от которых это время зависит:

$$\begin{aligned}
 t_{acc} = & (1 - p_{cr*})(p_{gc}(1 - p_{lc})p_{xcur}p_{rxcur}p_{mn}(t_{net} + t_{send} + t_{fp}) + \\
 & + p_{gc}(1 - p_{lc})p_{xcur}(1 - p_{rxcur})p_{mn}(t_{net} + t_{send} + t_{fp}) + \\
 & + p_{gc}(1 - p_{lc})p_{xcur}p_{rxcur}(1 - p_{mn})(2t_{net} + t_{send} + t_{fp}) + \\
 & + p_{gc}(1 - p_{lc})p_{xcur}(1 - p_{rxcur})(1 - p_{mn})(2t_{net} + t_{send} + t_{fp}) + \\
 & + p_{gc}p_{lc}p_{scur}p_{rxcur}p_{mn}2t_{net} + p_{gc}(1 - p_{lc})p_{scur}p_{rxcur}p_{mn}(t_{net} + t_{send}) + \\
 & + p_{gc}(1 - p_{lc})p_{scur}(1 - p_{rxcur})p_{mn}(t_{net} + t_{send}) + \\
 & + p_{gc}(1 - p_{lc})p_{scur}p_{rxcur}(1 - p_{mn})(2t_{net} + t_{send}) + \\
 & + p_{gc}(1 - p_{lc})p_{scur}(1 - p_{rxcur})(1 - p_{mn})(2t_{net} + t_{send}))
 \end{aligned} \tag{1}$$

где  $p_{mn}$  – вероятность того, что нода выполняющая запрос является нодой-владельцем (мастер)

$t_{net}$  – среднее время пересылки служебного сетевого пакета между нодами;

$t_{send}$  – время пересылки, включая время записи в память страницы от одной ноды к другой;

$t_{io}$  – время чтения блока с диска;

$p_{cr*}$  – вероятность того, что запрос обработается по  $scr^*$  сценарию;

$p_{rxcur}$  – вероятность того, что запрос будет XCUR;

$p_{xcur}$  – вероятность того, что состояние запрошенного блока XCUR;

$t_{fp}$  – время ожидания завершения обработки, из-за конфликта за блок

$p_{gc}$  – вероятность попадания в глобальный кэш;

$p_{out}$  – вероятность того, что блок оказался в состоянии «вылетел»;

$p_{lc}$  – вероятность попадания в локальный для обрабатывающей запрос ноды кэш;

$p_{down}$  – вероятность понижения блокировки XCUR при SCUR запросе.

Понижение блокировки, хотя и не присут-

ствует в формуле времени отклика (1), однако влияет на вероятности  $p_{xcur}$  и  $p_{scur}$ .

Первые четыре из 12-ти мы классифицировали как базовые характеристики системы,  $p_{rxcur}$  и  $p_{cr*}$  – как базовые характеристики трафика, а остальные как производные параметры.

Для простоты восприятия формулы (1) приведем 2 таблицы, полученные в [1] для не CR\* подтрафика.

В первой будут времена обработки соответствующих сценариев, а во второй вероятности попадания в эти сценарии, в зависимости от параметров приведенных выше.

**Табл.1. Времена обработки сценариев**

Запрос состоя- ния		Существует в глобальном кэше				вылетел	нет
		хсиг		Scur			
		л	г	л	г		
Мастер	хсиг	0	$t_{net}+t_{send}+t_{fp}$	0	$t_{net}+t_{send}$	$2t_{net}+t_{io}$	$t_{io}$
	ссиг	-(0)	$t_{net}+t_{send}+t_{fp}$	0	$t_{net}+t_{send}$		
Мастер	хсиг	0	$2t_{net}+t_{send}+t_{fp}$	$2t_{net}^*$	$2t_{net}+t_{send}^*$	$4t_{net}+t_{io}$	$2t_{net}+t_{io}$
	ссиг	-(0)	$2t_{net}+t_{send}+t_{fp}$	0	$2t_{net}+t_{send}$		

**Табл. 2. Вероятности попадания в сценарий.**

Запрос состоя- ния		Существует в глобальном кэше				вылетел	нет
		хсcur		сcur			
		л	г	л	г		
мастер	хсcur	$p_{gc}p_{lc}p_{xcur}$ $p_{rxcur}p_{mn}$	$p_{gc}(1-p_{lc})p_{xcur}$ $p_{rxcur}p_{mn}$	$p_{gc}p_{lc}p_{scur}$ $p_{rxcur}p_{mn}$	$p_{gc}(1-p_{lc})p_{scur}$ $p_{rxcur}p_{mn}$	$(1-p_{gc})p_{oui}p_{mn}$	$(1-p_{gc})(1-p_{oui})p_{mn}$
	сcur	$p_{gc}p_{lc}p_{xcur}$ $p_{rscur}p_{mn}$	$p_{gc}(1-p_{lc})p_{xcur}$ $p_{rscur}p_{mn}$	$p_{gc}p_{lc}p_{scur}$ $p_{rscur}p_{mn}$	$p_{gc}(1-p_{lc})p_{scur}$ $p_{rscur}p_{mn}$		
Не мастер	хсcur	$p_{gc}p_{lc}p_{xcur}$ $p_{rxcur}(1-p_{mn})$	$p_{gc}(1-p_{lc})p_{xcur}$ $p_{rxcur}(1-p_{mn})$	$p_{gc}p_{lc}p_{scur}$ $p_{rxcur}(1-p_{mn})$	$p_{gc}(1-p_{lc})p_{scur}$ $p_{rxcur}(1-p_{mn})$	$(1-p_{gc})p_{out}(1-p_{mn})$	$(1-p_{gc})(1-p_{out})(1-p_{mn})$
	сcur	$p_{gc}p_{lc}p_{xcur}$ $p_{rscur}(1-p_{mn})$	$p_{gc}(1-p_{lc})p_{xcur}$ $p_{rscur}(1-p_{mn})$	$p_{gc}p_{lc}p_{scur}$ $p_{rscur}(1-p_{mn})$	$p_{gc}(1-p_{lc})p_{scur}$ $p_{rscur}(1-p_{mn})$		

### Описание простейшего трафика для моделирования

Задачей данной работы будет привести примеры трафиков, на которых показать применимость модели вычисления времени отклика в кластерной системе. Или, перефразируя: задав базовые параметры трафика и системы рассчитать производные параметры, после чего получить среднее время отклика. Оценивая трафики в реальных системах стоит ориентироваться на

синтетические тесты, в частности на TCP-C [5]. В свою очередь трафик TPC-C распадается на подтрафики: транзакции нового заказа (New-Order, no) (44-45%), транзакции платежей (payment, pa), транзакции проверки статуса (Order-Status, os), транзакции сопровождения (delivery, de), Транзакции проверки склада (Stock-Level, sl). Отразим это в таблице 3.

**Табл. 3. Распределение подтрафиков  
TPC-C**

Transaction Type	Minimum % of mix
New-Order	n/a
Payment	43.0
Order-Status	4.0
Delivery	4.0
Stock-Level	4.0

Однако, задача математического моделирования трафика TPC-C теста достаточно громоздка. Максимально упростим его и создадим первичную модель трафика для применения математической модели. Такой подход также позволит исследовать кластерную систему. Назовём этот трафик простейшим. Впоследствии, после исследования простейшего трафика мы сможем добавить в него подтрафики, получая уже другой трафик и повторяя исследования для вновь полученного трафика. При добавлении в систему нового подтрафика, необходимо учитывать влияние добавленного подтрафика на трафик до добавления. Т. е. вводя новые подтрафики в систему мы влияем на ранее исследованные трафики. Идеальным случаем, когда влияние подтрафиков друг на друга сведено к нулю – это непересекаемость по общим ресурсам. Однако, уже моделирование простейшего трафика позволит показать применимость модели, а последующее усложнение простейшего трафика до TPC-C трафика, позволит увидеть применимость модели и для среднестатистической OLTP системы.

Проанализировав профили транзакций и соотношения в таблице 3, можно выявить, что наиболее интенсивный трафик – это трафик генерируемый транзакцией по (новый заказ) и в этом подтрафике преобладает вставка строк. Учитывая то, что в оракловой реализации TPC-C трафика целесообразно использовать последовательности и неблокирующий SELECT (SELECT FOR UPDATE) логичной при построении простейшего трафика будет следующая постановка задачи:

Пусть мы имеем 2 таблицы: таблица поиска (search s) и таблица проводок (posting p). Клиент сначала ищет по критериям продукт в таблице поиска и его наличие на складе ( $S_s, S_p$ ), потом эксклюзивно блокирует количество продукта на складе и перепроверяет наличие найденного продукта на складе т.к. за время чтения результатов предыдущего запроса продукт мог закончиться ( $XS_s, S_p$ ), затем в

случае наличия выполняет проводку ( $I_p$ ) и завершает транзакцию (C). т.е.

$$S_s S_p \Rightarrow XS_s S_p \Rightarrow I_p \Rightarrow C$$

$S$  – выборка из двух таблиц. В первой – параметры выборки и ID, во второй проводки. Все блоки запрашивают SCUR.

$XS_s$  – один блок запрашивает XCUR

$S_p$  запрашивает строки (может быть несколько блоков) на чтение SCUR по ID (доступ через индекс)

$I_p$  – запрашивает один блок

Таким образом, мы получили строгое описание простейшего трафика. Отметим, что простейший трафик состоит только из SCUR и XCUR запросов. Это предполагает отсутствие в трафике подтрафика  $CR^*$ , т.е система имеет только один базовый параметр трафика. Это  $p_{rxcur}$  – определяющий соотношение между SCUR и XCUR запросами.

### Определение производных параметров

Зададим, что облачная система состоит из  $n$  нод и все ноды одинаковой ёмкости. Тогда  $p_{mn}=1/n$ . Также будем считать, что размер глобального кэша достаточно большой, чтобы закэшировать все необходимые данные. Тогда на нахождение в локальном кэше будет влиять лишь то, остался ли блок в локальном кэше в нужном состоянии с момента последней обработки данных или нет. Очевидно, что если в системе нет XCUR блокировок, то через некоторое время все запрашиваемые блоки будут в локальных кэшах. Это означает, что  $p_{gc}=1$ , а  $p_{out}=0$ . Оценивая современные конфигурации облачных OLTP систем, чтения с диска составляют небольшой процент чтений относительно общего для оптимально настроенных имеющимися инженерными средствами систем. Также, есть тенденция по перенесению всей активной части БД в память [7,8]. Для упрощения моделирования будем считать, что коллапса по блокировкам нет [2].

Таким образом:  $p_{mn}=1/n$ ,  $p_{gc}=1$ ,  $t_{fp}=0$ . Как было сказано выше, в простейшем трафике подтрафик  $CR^*$  отсутствует, а значит  $p_{cr}=0$ . Исходя из итоговой формулы (1), если  $p_{gc}=1$ , параметры  $p_{out}$ ,  $t_{io}$  не влияют на конечное время отклика. Значит, задача в том, чтобы для простейшего трафика установить зависимость между базовой характеристикой трафика  $p_{rxcur}$ , базовыми характеристиками системы  $t_{net}$  и  $t_{send}$  и производными параметрами  $p_{xcur}$ ,  $p_{lc}$  и  $p_{down}$ .

### Определение производных параметров для трафика к таблице поиска

Для таблицы поиска имеем последовательность  $S_s \Rightarrow XS_s \Rightarrow C$ . При этом первый запрос запрашивает на блок состояние SCUR, а второй запрос XCUR. В таком случае, вероятности  $p_{rxcur}$  и  $p_{rscur}$  распределяются равномерно:  $p_{rxcur}=p_{rscur}=1/2$ . Так как имеется связь между состояниями имеет смысл рассмотреть сценарии смены состояний  $xs$ ,  $ss$ ,  $sx$ ,  $xx$ , а не вероятности нахождения в состояниях. Поясним значения этих аббревиатур:  $xs$  это сценарий смены состояния с XCUR на SCUR,  $xx$  – с XCUR на XCUR – т. е. состояние не изменилось и т.д.. Поскольку  $p_{rxcur}=1/2$  и в результате запроса состояния XCUR будет обрабатываться либо  $sx$  либо  $xx$  сценарий, получаем  $p_{xx}+p_{sx}=1/2$ . Аналогично  $p_{xs}+p_{ss}=1/2$ . Пусть вероятность  $p_{xx}=\beta_1$ , а вероятность  $p_{ss}=\beta_2$ . В ситуации, если конфликтов за блок не возникает, мы будем иметь только последовательность смены состояний  $xs \Rightarrow sx$  и  $sx \Rightarrow xs$ . В случае, если две или более сессий одновременно работают с блоком – возникают  $xx$  и  $ss$  сценарии. Причем количество  $xx$  будет равно количеству  $ss$  т. к. все последовательности запросов " $S_s \Rightarrow XS_s \Rightarrow C$ " завершаются – т.е. за каждым  $S_s$  следует  $XS_s$ . В трафиках, где не все последовательности завершаются – значения  $\beta_1$  и  $\beta_2$  различные. Тогда:

$$\begin{aligned} p_{xcur}p_{rxcur} &= p_{xx} = \beta_k p_{scur}p_{rscur} = p_{ss} = \\ \beta_k p_{scur}p_{rxcur} &= p_{sx} = 1/2 - \beta_k p_{xcur}p_{rscur} = \\ p_{xs} &= 1/2 - \beta_k \quad (2) \end{aligned}$$

индекс  $k$  в данном случае показывает, что эти вероятности меняются для каждого  $k$ -того блока. Подставив в таблицу формулы (2) и зная, что вероятность локального кэша  $p_{lc}=1/n$  имеем:

**Табл. 4. Распределение вероятностей  
в подтрафике поиска**

Запрос состоя- ния		Находится в состоянии			
		xcur		scur	
		л	г	л	г
Мастер	xcur	$\beta_k/n^2$	$\beta_k(1-1/n)/1/n$	$(1/2-\beta_k)/1/n^2$	$(1/2-\beta_k)/(1-1/n)1/n$
	scur	$(1/2-\beta_k)/1/n^2$	$(1/2-\beta_k)/(1-1/n)1/n$	$\beta_k/n^2$	$\beta_k 1/n/(1-1/n)$
Не мастер	xcur	$\beta_k(1-1/n)/1/n$	$\beta_k(1-1/n)^2$	$(1/2-\beta_k)/(1-1/n)1/n$	$(1/2-\beta_k)/(1-1/n)^2$
	scur	$(1/2-\beta_k)/1/n(1-1/n)$	$(1/2-\beta_k)/(1-1/n)^2$	$\beta_k 1/n/(1-1/n)$	$\beta_k(1-1/n)^2$

**Табл. 5. Времена обработки сценариев  
в подтрафике поиска**

Запрос состоя- ния		Находится в состоянии			
		xcur		scur	
		л	г	л	г
Мастер	xcur	0	$t_{net}+t_{send}$	0	$t_{net}+t_{send}$
	scur	0	$t_{net}+t_{send}$	0	$t_{net}+t_{send}$
Не мастер	xcur	0	$2t_{net}+t_{send}$	$2t_{net}$	$2t_{net}+t_{send}$
	scur	0	$2t_{net}+t_{send}$	0	$2t_{net}+t_{send}$

Заметим, что время отклика явно не зависит от интенсивности, однако увеличение интенсивности повышает вероятность конфликта за блок даже в случаях когда конфликта за ресурс нету. Выведем формулу времени доступа к блоку в зависимости от  $\beta_k$ .

$$\begin{aligned} \frac{t_{acc}}{t_{send}} &= \beta_k \left(1 - \frac{1}{n}\right) \frac{1}{n} \left(1 + \frac{t_{net}}{t_{send}}\right) + \\ &+ \left(\frac{1}{2} - \beta_k\right) \left(1 - \frac{1}{n}\right) \frac{1}{n} \left(1 + \frac{t_{net}}{t_{send}}\right) + \\ &+ \beta_k \left(1 - \frac{1}{n}\right)^2 \left(1 + 2 \frac{t_{net}}{t_{send}}\right) + \\ &+ \left(\frac{1}{2} - \beta_k\right) \left(1 - \frac{1}{n}\right)^2 \left(1 + 2 \frac{t_{net}}{t_{send}}\right) + \\ &+ 2 \frac{t_{net}}{t_{send}} \left(\frac{1}{2} - \beta_k\right) \frac{1}{n} \left(1 - \frac{1}{n}\right) + \\ &+ \beta_k \left(1 - \frac{1}{n}\right) \frac{1}{n} \left(1 + \frac{t_{net}}{t_{send}}\right) + \\ &+ \left(\frac{1}{2} - \beta_k\right) \left(1 - \frac{1}{n}\right) \frac{1}{n} \left(1 + \frac{t_{net}}{t_{send}}\right) + \\ &+ \left(\frac{1}{2} - \beta_k\right) \left(1 - \frac{1}{n}\right)^2 \left(1 + 2 \frac{t_{net}}{t_{send}}\right) + \\ &+ \beta_k \left(1 - \frac{1}{n}\right)^2 \left(1 + 2 \frac{t_{net}}{t_{send}}\right) = \\ &= \left(1 - \frac{1}{n}\right) \frac{1}{n} \left(1 + \frac{t_{net}}{t_{send}}\right) + \\ &+ \left(1 - \frac{1}{n}\right)^2 \left(1 + 2 \frac{t_{net}}{t_{send}}\right) + \\ &+ 2 \left(\frac{1}{2} - \beta_k\right) \frac{1}{n} \left(1 - \frac{1}{n}\right) \frac{t_{net}}{t_{send}} \end{aligned}$$

Преобразовав, получаем конечную формулу, отражающую линейную зависимость от  $\beta_k$ :

$$\begin{aligned} t_{acc} &= \left( \left(1 - \frac{1}{n}\right) \left(1 + 2 \frac{t_{net}}{t_{send}}\right) - 2\beta_k \frac{1}{n} \left(1 - \frac{1}{n}\right) \frac{t_{net}}{t_{send}} \right) t_{send} \quad (3) \end{aligned}$$



Интересно заметить, что конфликт за блоки ( $\beta_k > 0$ ) без конфликта за ресурсы не увеличивает, а уменьшает время доступа к блоку. Это связано с тем что время выполнения сценариев  $ss+xx$  меньше чем время выполнения  $sx+xs$  в распределённой системе за счет нулевого времени доступа при выполнении  $ss$  сценария при наличии SCUR блока на локальной ноде.

Рассматривая индексы таблицы поиска можно констатировать, что имеют место только SCUR запросы. Целесообразно считать, что все блоки индекса находятся в локальной памяти, поскольку запрашиваются постоянно и только на чтение, т. к. таблица поиска не изменяется. Соответственно, время доступа к ним  $t_{ac.index}$  нулевое. Количество читаемых при этом индексных блоков, при условии что индекс без перекосов – т. е. регулярно перестраивается, равно высоте индекса. Обозначим её  $h$ . Поскольку в каждом обращении через индекс таблицы поиска делается 2 запроса, несложно прийти к выводу, что на каждое обращение читается  $2h$  индексных блоков. Т.е.  $t_{ac.index} = 0$ ,  $bl_{index} = 2h$ .

### Определение производных параметров для трафика к таблице проводок

Ситуация для таблицы проводок несколько иная. Таблица проводок имеет один индекс – индекс первичного ключа. Если таблица проводок будет организована как IOT [6], то это позволит работать только с блоками индекса. Примем такой подход, поскольку он позволит сократить количество вставок и упростит анализ. Для индексных блоков (блоков ИОТ) вероятность запроса блока на изменение (XCUR) другими нодами зависит от количества строк в блоке и количества нод. Исходя из статистики интенсивности изменения и статистики интенсивности чтения индексного блока (или соотношения XCUR к SCUR запросов к блоку) можно определить оптимальное значение  $p_{down}$ . Могут быть несколько подходов, например по заполненности индексного блока, или оценивая статистику соотношения XCUR и SCUR запросов, что применяется Oracle [4]. Объективно, необходимо вычисляя оптимальное значение  $p_{down}$ , понижать или не понижать эту блокировку, однако для упрощения формул будем считать  $p_{down} = 1$  для индексных блоков. Поиск оптимального значения  $p_{down}$ , а также алгоритм понижения блокировки могут быть предметом

оптимизации, однако это выходит за рамки приведенной статьи. Также важной характеристикой, влияющей на  $p_{rxcur}$ ,  $p_{xcur}$ ,  $p_{lc}$ , является «страничность» данных: количество строк в одном блоке. Для того чтобы задать страничность, обозначим  $\alpha$  величину обратную количеству строк в блоке. Можно заметить, что задача опять свелась к тому, чтобы для вывести соотношение между параметрами подтрафика проводок  $p_{rxcur}$ ,  $p_{xcur}$ ,  $p_{lc}$  с учётом базовых характеристик  $t_{net}$  и  $t_{send}$ , но уже для таблицы организованной как IOT.

Распишем простейший трафик для таблицы проводок. Будем рассматривать таблицу как набор ресурсов. ID ресурса в таблице поиска (номер клиента, номер продукта и т. д.) будет и идентификатором ресурса в таблице проводок. Пусть интенсивность заявок на доступ к  $j$ -тому ресурсу  $v_j$ . Тогда интенсивность XCUR запросов к  $j$ -му будет  $v_j + [\alpha v_j]$ , поскольку в каждой заявке к таблице проводок присутствует один запрос –  $I$  и этот запрос будет изменять как листовые [6] так и промежуточные [6] блоки. Величина  $[\alpha v_j]$ , здесь отражает интенсивность изменения промежуточных блоков (интенсивностью изменения промежуточных блоков уровней выше первого пренебрегаем).  $v_j$  – интенсивность изменения листовых блоков. Учитывая то, что листовые блоки изменяются только при появлении нового листового блока, дискретность модели задаётся с помощью функции антье, обозначенной квадратными скобками. Имеет смысл  $v_{rxcur(j)}$  разделить трафик к промежуточным (branch) блокам IOT  $v_{rxcur.b(j)}$  и к листовым (leaf)  $v_{rxcur.l(j)}$ :

$$v_{rxcur.l(j)} = v_j \quad v_{rxcur.b(j)} = [\alpha v_j]$$

Трафик к промежуточным блокам – это тот же трафик только уменьшенный пропорционально количеству строк (выражается умножением на  $\alpha$ ). Таким образом, последовательность ( $S_s \ S_p \Rightarrow X S_s \ S_p \Rightarrow I_p \Rightarrow C$ ) распадается на 4 подтрафика не конфликтующих за общие блоки, а значит и за состояния SCUR либо XCUR). Это подтрафики: к листовым блокам организованной как индекс таблицы (IOT) проводок, к промежуточным блокам таблицы, к таблице поиска, также подтрафики к индексам таблицы поиска.

Начнём работать с подтрафиком к листовым блокам. Поскольку  $v_{rxcur.l(j)}$  – интенсивность запросов блока в состоянии XCUR, и интенсивность – это количество добавленных строк (один  $I_p$  на проводку) отнесённое ко

времени и в блоке  $1/\alpha$  строк, то

$$v_{rxcur.l(j)} = \frac{\frac{1}{\alpha} db l_{xcur.l(j)}}{dt} = v_j$$

где  $bl_{xcur.l}$  – количество листовых блоков с состоянием XCUR. Тогда, с учетом дискретности, зависимость XCUR листовых блоков от времени:

$$bl_{xcur.l} = \left[ \alpha \int_0^t v_{rxcur.l(j)} dt \right] + 1 = [\alpha v_j t] + 1$$

Поскольку все блоки  $j$ -го ресурса, на которые ранее запрашивался XCUR, в следующем запросе будут запрашиваться SCUR, имеем

$$bl_{scur.l} = \left[ \alpha \int_0^t v_{rxcur.l(j)} dt \right] + 1 = [\alpha v_j t] + 1$$

т.к. при работе с листовыми блоками все блоки надо прочитать во время проводки дважды ( $S_p \Rightarrow S_p \Rightarrow I_p$ ) имеем:

$$\begin{aligned} v_{rscur.l(j)} &= 2v_j; \quad t \in (0, \frac{1}{\alpha v_j}] \\ v_{rscur.l(j)} &= 4v_j; \quad t \in (\frac{1}{\alpha v_j}, \frac{2}{\alpha v_j}] \\ v_{rscur.l(j)} &= 6v_j; \quad t \in (\frac{2}{\alpha v_j}, \frac{3}{\alpha v_j}] \\ &\dots \\ v_{rscur.l(j)} &\approx 2\alpha v_j^2 t; \quad t \in (\frac{10}{\alpha v_j}, \infty] \end{aligned}$$

На последнем диапазоне можем применять непрерывную модель, поскольку относительная погрешность не превышает 10%.

Тогда распишем вероятности запрашивания для диапазона  $(10/\alpha v_j t, \infty)$

$$\begin{aligned} p_{rxcur(j)} &= \frac{v_{rxcur}}{(v_{rscur} + v_{rxcur})} \\ p_{rxcur(j)} &= \frac{v_j}{v_j + 2\alpha v_j^2 t} = \frac{1}{(1 + 2\alpha v_j t)} \end{aligned}$$

для дискретной модели:

$$p_{rxcur(j)} = \frac{1}{(1 + 2[\alpha v_j t])}$$

Аналогично

$$\begin{aligned} p_{rscur(j)} &= 1 - p_{xscur(j)} \\ p_{rscur(j)} &= 1 - \frac{1}{1 + 2v_j t} = \frac{2v_j t}{1 + 2v_j t} \\ p_{rscur(j)} &= \frac{2[\alpha v_j t]}{1 + 2[\alpha v_j t]} \end{aligned}$$

Распишем теперь вероятности состояния блоков таблицы проводок:

Поскольку во всей последовательности ( $S_s S_p$

$\Rightarrow X S_s S_p \Rightarrow I_p \Rightarrow C$ ) только одна операция по таблице проводок, требующая XCUR ( $I_p$ ), интенсивность  $xscur$  равна:  $v_{rxcur(j)} = v_j$

А так, как на одну строку в состоянии XCUR при выполнении всей последовательности действий по таблице проводок приходится  $v_j t$  строк (в данном случае и блоков) в состоянии SCUR, то получаем вероятности:

$$p_{xcur(j)} = \frac{1}{1 + [2\alpha v_j t]} (4) \quad p_{scur(j)} = \frac{[2\alpha v_j t]}{1 + [2\alpha v_j t]} (5)$$

Вероятность мастер-ноды, если количество нод  $n$   $p_{mn} = 1/n$ .

Распишем вероятность  $p_{lc}$ . Отметим, что она состоит из 4х взаимно исключающих вероятностей  $p_{lcxx}$ ,  $p_{lcxx}$ ,  $p_{lcxx}$ ,  $p_{lcxx}$ . Принципиально, зная вероятности нахождения блока в  $xscur$  и  $scur$  состояниях и вероятности запросов  $xscur$  и  $scur$  блокировок можно с учётом равномерности запросов по нодам путём перемножения вероятностей получить  $p_{lcxx}$ ,  $p_{lcxx}$ ,  $p_{lcxx}$ ,  $p_{lcxx}$ . Однако, перемножать вероятности можно лишь в том случае если события независимы. В нашем случае это не так, и придётся отбросив формулы вероятностей нахождения блока (3,4), полученные выше, на основе данных о трафике вычислить  $p_{lcxx}$ ,  $p_{lcxx}$ ,  $p_{lcxx}$ ,  $p_{lcxx}$ .

Вероятность  $p_{lcxx}$  (та которая в таблице распределения вероятностей будет перемножаться с  $p_{xcur}$  и  $p_{rxcur}$ ) – это вероятность такой последовательности чтений блока, при котором блок эксклюзивно (XCUR) был запрошен этой (не другими) нодой ранее. Однако в случае простейшего трафика, в момент проводки  $j$ -го ресурса другие последовательности команд трафика блокируются при эксклюзивном чтении таблицы поиска ( $X S_s$ ). Если предположить диапазоны ИОТ для разных  $j$  (ресурсов) не пересекаются то, для данного трафика  $p_{lcxx} = 0$ .

Введём условие неконфликтности. Т.е. Пусть модель применяется в таких диапазонах значений, когда за время обработки заявки вероятность конфликта настолько мала, что временем затраченным на разрешение конфликта можно пренебречь. Это в свою очередь накладывает ограничения на интенсивность и время обработки заявки ( $S_s S_p \Rightarrow X S_s S_p \Rightarrow I_p \Rightarrow C$ ):  $t \ll 1/v_j$ . Хотя такое соотношение и не учитывает задержек связанных с конфликтом, будем предполагать, что разрешение конфликта требует не более 10 времен обычной обработки заявки. Тогда  $t \ll 10/v_j$  или  $v_j t_{ac} < 100$ . Это в принципе очень не верно, т. к. всегда между  $S_s S_p \Rightarrow X S_s S_p$  есть

время  $think\_time$  – т. е. время обдумывания оператором полученного результата на первых двух запросах. Однако, на определённых системах может быть минимизировано  $think\_time$  за счёт кэширования активной части БД на серверах приложений. Для неактивной части БД (а неактивная часть БД это  $j$ -й ресурс с низкой интенсивностью  $v_j$ ) большое время  $think\_time$  не приводит к конфликтам. Таким образом, если считать что интенсивность к каждому ресурсу такова, что время обработки заявки (с учётом времени доступа к блоку, которое мы исследуем)  $S_s S_p \Rightarrow X S_s S_p \Rightarrow I_p \Rightarrow C \ll 1/v_j$  то конфликтов не возникает.

В итоговой формуле необходимо будет неравенство  $v_j t_{ac} < 100$  применить. Таким образом, условие неконфликтности существенно упрощает анализ. Стоит, однако заметить, что условие не является критическим для применения модели, и влияет всего лишь на распределение вероятностей и времена обработок соответствующим таким вероятностям. Более того при анализе устойчивости к коллапсу по блокировкам пренебрегать вероятностями конфликта и распределением вероятностей разных сценариев обработки блока нельзя. Однако на данном шаге, где ставится задача показать применимость модели – это допустимо.

Вероятности  $p_{lcss}$ ,  $p_{lcss}$ ,  $p_{lcss}$ : рассчитываем исходя из заблокированности таблицы поиска и условия неконфликтности. Это значит, что первый запрос  $S_p$  может читать блоки как с удалённых нод так и с локальной. При этом только один блок может быть прочитан XCUR с удалённой ноды. Остальные будут читаться локально.

$$p_{ss} = \frac{2([av_j t] + 1) - 1}{1 + 2([av_j t] + 1)} = \frac{2[av_j t] + 1}{2[av_j t] + 3} (6) p_{xs} = p_{sx} = \frac{1}{1 + 2([av_j t] + 1)} = \frac{1}{2[av_j t] + 3} \quad (7)$$

Каждая из вероятностей  $p_{ss}$ ,  $p_{sx}$ ,  $p_{xs}$  распадается на глобальную и локальную вероятности, соответствующие событиям чтения блока (страницы) из локального либо глобального но не локального кэша. Вопрос стоит в распределении и  $p_{ss}$ ,  $p_{sx}$ ,  $p_{xs}$  между своими локальными и глобальными составляющими.

Начнём с  $p_{xs}$ .  $p_{xs}$  – распределяется как  $p_{lxs} = 1$  и  $p_{gxs} = 0$ , поскольку XCUR запрос в этом трафике, выполняемый командой  $I_p$ , всегда происходит после SCUR запроса  $S_p$ . Причем блокировка  $S_s$  гарантирует, что другие ноды с момента запроса SCUR блокировки запросом  $S_p$  не

выполняли XCUR запросов на таблицу проводок, а значит при XCUR после SCUR блок всегда в локальном кэше.

$p_{sx}$ : поскольку условие неконфликтности предполагает, что за время выполнения последовательности команд другие ноды не выполняют XCUR запросы на данный ресурс. А это означает, что тот блок, который может измениться (последний блок) в последний раз изменялся либо локальной нодой либо всеми остальными. В таком случае  $p_{lsx} = 1/n$  и  $p_{gsx} = (1 - 1/n)$

$p_{ss}$  – распределяется более сложно:

Количество заполненных блоков  $m = [av_j t] + 1$ ,  
Время между заявками на одной ноды  $1/(nv_j)$

Количество изменённых с прошлой обработки на ноды блоков, т. е. сколько блоков требуется прочитать из глобального кэша  $[av_j t] - [av_j(t - 1/(nv_j))]$ . Поскольку второй раз они уже в локальном кэше то общее количество блоков  $2m = 2[av_j t]$ . Тогда  $p_{ss}$  распределяется между локальным и глобальным кешем в диапазоне  $t > 1/(nv_j)$  как:

$$p_{gss} = \frac{[av_j t] - [av_j(t - \frac{1}{nv_j})]}{2([av_j t] + 1)}$$

$$p_{lss} = 1 - \frac{[av_j t] - [av_j(t - \frac{1}{nv_j})]}{2([av_j t] + 1)} = \frac{[av_j t] + [av_j(t - \frac{1}{nv_j})] + 2}{2([av_j t] + 1)}$$

или

$$p_{gcss} = \frac{[av_j t] - [av_j t - \frac{\alpha}{n}]}{2([av_j t] + 1)} \cdot \frac{2[av_j t] + 1}{2[av_j t] + 3} \quad (8)$$

$$p_{lcss} = \left( \frac{[av_j t] + [av_j t - \frac{\alpha}{n}] + 2}{2([av_j t] + 1)} \right) \cdot \frac{2[av_j t] + 1}{2[av_j t] + 3} \quad (9)$$

аналогично для диапазона  $t < 1/(nv_j)$  – что отражает первую обработку на ноды  $j$ -того ресурса:

$$p_{gss} = \frac{[av_j t]}{2[av_j t]} = \frac{1}{2} \quad p_{lss} = 1 - \frac{[av_j t]}{2[av_j t]} = \frac{1}{2} \quad (10)$$

$$p_{lcss} = p_{gcss} = \frac{2([av_j t] + 1) - 1}{2 + 4([av_j t] + 1)} \quad (11)$$

учитывая, что  $p_{xx} = 0$  и  $p_{lxs} = 0$ , а также подставив полученные в формулах выше (6,7,8,9,10,11) величины получаем видоизменённые таблицы:

**Табл.5. Распределение вероятностей в подтрафике проводок**

Запрос состо- яния		Находится в состоянии			
		хсиг		сиг	
мастер	л	г	л	г	
	хсиг	г	л	г	
не мастер	хсиг	г	л	г	
	сиг	г	л	г	
мастер	хсиг	0	0	$p_{sx}/n$	0
	сиг	$p_{xs}/n^2$	$p_{xs}(1-1/n)/n$	$p_{lcss}/n$	$p_{gcsc}/n$
не мастер	хсиг	0	0	$p_{sx}(1-1/n)$	0
	сиг	$(p_{xs}/n)/(1-1/n)$	$p_{xs}(1-1/n)^2$	$p_{lcss}(1-1/n)$	$p_{gcsc}/(1-1/n)$

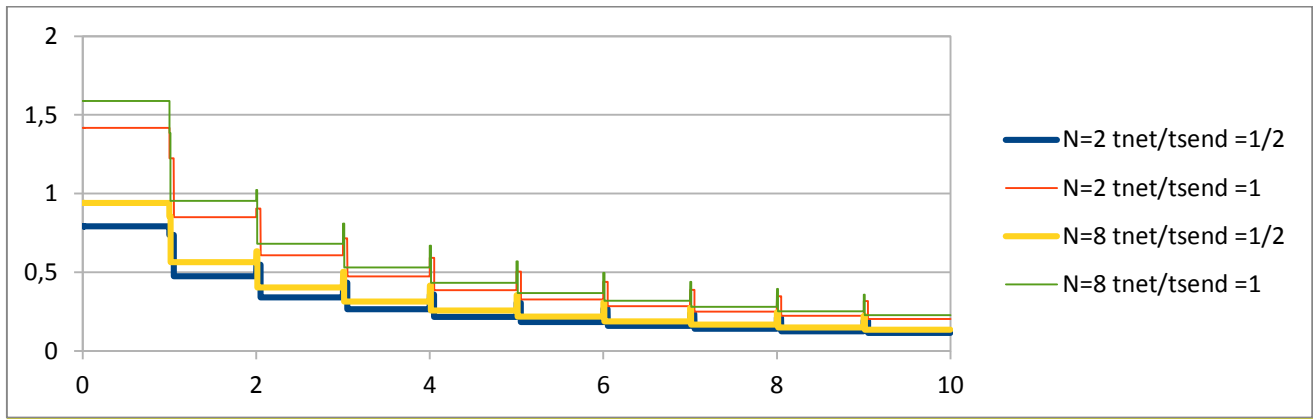
**Табл. 6. Времена обработки сценариев в подтрафике проводок**

Запрос состо- яния		Находится в состоянии			
		хсиг		сиг	
мастер	л	г	л	г	
	хсиг	г	л	г	
не мастер	хсиг	0	$t_{net}+t_{send}$	0	$t_{net}+t_{send}$
	сиг	-(0)	$t_{net}+t_{send}$	0	$t_{net}+t_{send}$
мастер	хсиг	0	$2t_{net}+t_{send}$	$2t_{net}^*$	$2t_{net}+t_{send}^*$
	сиг	-(0)	$2t_{net}+t_{send}$	0	$2t_{net}+t_{send}$

$$\begin{aligned}
 \frac{t_{ac}}{t_{send}} &= \left( p_{xs} \left( 1 - \frac{1}{n} \right) \frac{1}{n} \right) \left( 1 + \frac{t_{net}}{t_{send}} \right) + \\
 &+ \left( p_{xs} \left( 1 - \frac{1}{n} \right)^2 \right) \left( 1 + 2 \frac{t_{net}}{t_{send}} \right) + \\
 &+ \frac{p_{gcsc}}{n} \left( 1 + \frac{t_{net}}{t_{send}} \right) + \\
 &+ p_{gcsc} \left( 1 - \frac{1}{n} \right) \left( 1 + 2 \frac{t_{net}}{t_{send}} \right) = \\
 &= \left( p_{xs} \left( 1 - \frac{1}{n} \right) + p_{gcsc} \right) * \\
 &* \left( \frac{1}{n} + \frac{1}{n} \frac{t_{net}}{t_{send}} + \left( 1 - \frac{1}{n} \right) \left( 1 + 2 \frac{t_{net}}{t_{send}} \right) \right) \\
 &= \left( p_{sx} - \frac{p_{xs}}{n} + p_{gcsc} \right) \left( 1 + 2 \frac{t_{net}}{t_{send}} - \frac{1}{n} \frac{t_{net}}{t_{send}} \right)
 \end{aligned}$$

Получаем формулу  $t_{ac}$ . Также отразим  $t_{ac}$  от масштабированного времени ( $\alpha v_j t$ ). для различных комбинаций соотношений  $t_{net}/t_{send}$  и  $1/n$  на графике:

$$t_{ac} = \frac{\left( 1 + 2 \frac{t_{net}}{t_{send}} - \frac{1}{n} \frac{t_{net}}{t_{send}} \right)}{2[\alpha v_j t] + 3} \left( 1 - \frac{1}{n} + \frac{[\alpha v_j t] - [\alpha v_j t - \frac{\alpha}{n}]}{2([\alpha v_j t] + 1)} \cdot (2[\alpha v_j t] + 1) \right) t_{send} \quad (12)$$


**Рис. 1. График зависимости  $t_{ac}$  от масштабированного времени**

Всплески в окрестности целых чисел характеризуют скачкообразность увеличения количества обрабатываемых блоков, но каждый раз на 1.

Также приведём на графике время доступа к блоку по сводному трафику подтрафика поиска и подтрафика проводок. При формировании сводного времени отклика по двум подтрафикам необходимо учитывать что количественно блоков подтрафика поиска это всегда 2, а количество блоков подтрафика проводок  $2[\alpha v_j t] + 1$ . Таким образом:

$$t_{ac} = \frac{(2[\alpha v_j t] + 1)t_{acc.проводок} + 2t_{ac.поиска}}{(2[\alpha v_j t] + 3)}$$

Заметим, что  $t_{ac.поиска}$  зависит от еще и от третьего параметра  $\beta_k$ , и возьмём сначала значение  $\beta_k$  равные 0.

Напомним, что возможные значения  $\beta_k$  лежат в диапазоне  $[0, 1/2)$ . Приведём график времени доступа к блоку по сводному трафику по 2 трафикам: трафику по листовым блокам таблицы проводок и трафику по таблице поиска.



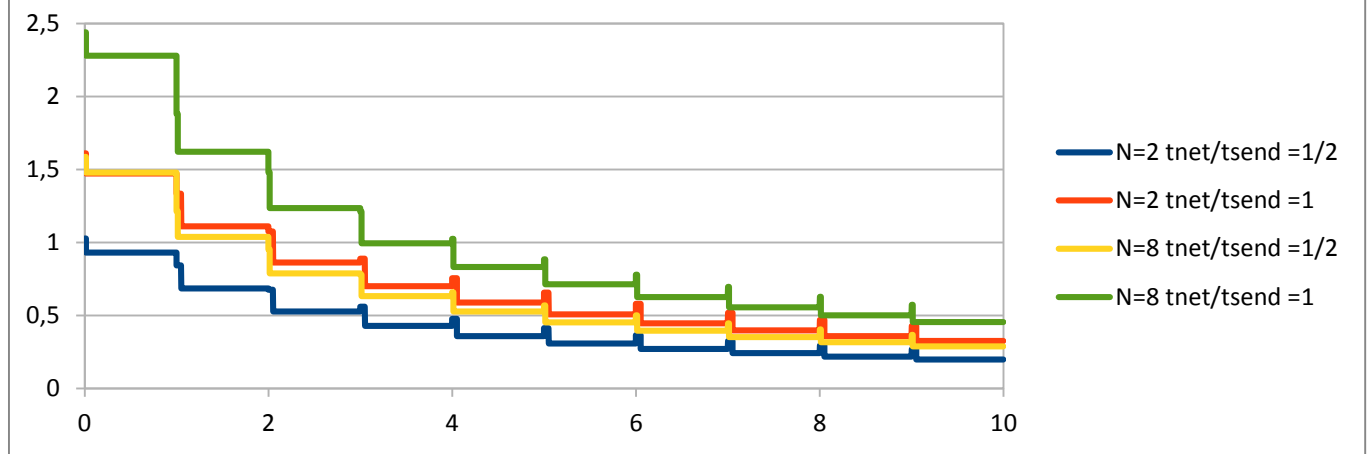


Рис. 2. Сводный трафик по листовым блокам ИОТ и блокам таблицы поиска

Поскольку зависимость от  $\beta_k$  линейная, то имеет смысл взять разницу времён доступа в крайних точках и отнести её к значению

большого времени доступа. Таким образом, мы получим погрешность, вносимую  $\beta_k$  в общее время доступа. Приведём на графике:

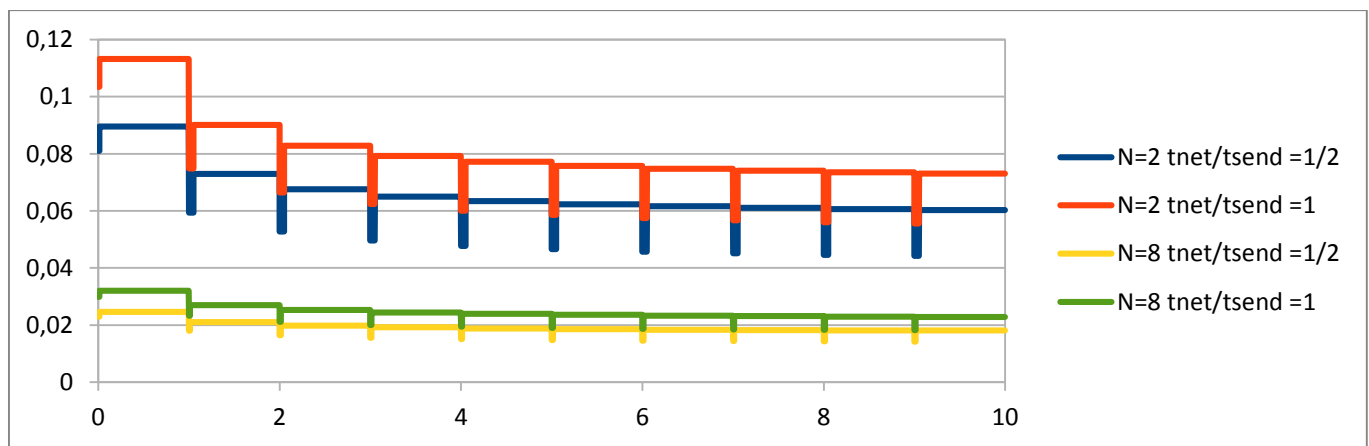


Рис. 3. График Относительная погрешность вносимая  $\beta_k$

Любопытно заметить, что для приведенных трафиков, кроме значения  $n=2$  и  $t_{net}/t_{send}=1$ , относительная погрешность менее 10%. Продифференцировав по  $n$ , а также продифференцировав по  $t_{net}/t_{send}$  можно показать, что на диапазонах значений  $n>2$  и  $t_{net}/t_{send}<1$  производная относительной погрешности как по  $n$  так и по  $t_{net}/t_{send}$  отрицательная. А это значит, что максимальное значение достигается в крайней точке  $n=2$  и  $t_{net}/t_{send}=1$ . т.е. относительная погрешность всегда менее 11.3 %. при этом стоит заметить что значение  $\beta_k=1/2$  соответствует ситуации когда количество конфликтов за блок бесконечность, что при условии неконфликтности за  $k$ -й ресурс реально ограничено количеством строк в блоке. Всё это создаёт предпосылки для пренебрежения влиянием параметра  $\beta_k$  на трафик и использовании упрощенных формул, предполагающих значение  $\beta_k=1/4$ , при котором относительная погрешность не превысит 5.7%.

Также интересно заметить, что  $\beta_k=1/4$  - это вероятность 1 конфликта за блок без конфликта за ресурс во время выполнения последовательности команд простейшего трафика.

С чтением промежуточных блоков индексов таблицы проводок ситуация сложнее. Интенсивность  $x_{sig}$  запросов  $\alpha v_j$ , при этом интенсивность  $s_{sig}$  запросов  $2(h-1)v_j$ . Считая что промежуточный трафик состоит только из  $s_{sig}$  запросов мы предполагаем что рассчитанное время доступа будет на порядок больше среднего времени обработки  $x_s$  и  $s_x$  запросов. Количество  $xx$  запросов при условии неблокирования, будет всегда выделяться из  $x_s$  запросов, при этом снижая количество  $s_x$  запросов – они станут  $ss$ . Таким образом минимальное количество  $ss$  запросов будет при непересекаемости ресурсов по блокам (отсутствие  $xx$  запросов). Задача поиска  $t_{ac.branch}$  может представлять академический интерес, однако для простоты вычислений

будем считать его пренебрежимо малым. Сформируем условия при которых допустимо пренебрегать им. Зная, что количество блоков читаемых при сценариях  $x_s$  или  $s_x$ :

$$bl_{sx} = bl_{xs} = \alpha v_j; \quad bl_{ss} = 2(h-1)v_j - \alpha v_j$$

расписываем вероятности  $s_x$  и  $x_s$ , аналогично формулам (6,7,8,9,10,11) выведенным выше при непересекаемости (отсутствии  $xx$  запросов):

$$p_{sx} = p_{xs} = \frac{\alpha v_j}{(2(h-1)v_j + \alpha v_j)} = \frac{\alpha}{(2(h-1) + \alpha)}$$

$$p_{ss} = \frac{\alpha v_j}{(2(h-1) - \alpha)} = \frac{\alpha}{(2(h-1) + \alpha)}$$

Таким образом, условие допустимости считать трафик к промежуточным блокам  $read$  only трафиком являются такие значения  $\alpha$  и  $h$ , при которых будет выполняться неравенство:

$$t_{ac} \gg \left( \frac{\alpha v_j}{(2(h-1)v_j + \alpha v_j)} \right) (2t_{net} + t_{send})$$

$$\Rightarrow t_{ac} \gg \left( \frac{\alpha \left( 2 \frac{t_{net}}{t_{send}} + 1 \right)}{(2(h-1) + \alpha)} \right) t_{send}$$

$$\left( \frac{\alpha \left( 2 \frac{t_{net}}{t_{send}} + 1 \right)}{t_{ac}(2(h-1) + \alpha)} \right) t_{send} \ll 1 \quad (13)$$

Если это время будет на порядок ниже рассчитанного времени доступа – то количество промежуточных блоков индексов

$$bl_{branch} = 2(h_{branch} - 1), \text{ а время } t_{ac.branch} = 0.$$

Для итогового трафика по всем 4м подтрафикам при вычислениях будем считать высоты всех индекса таблицы поиска равные 3, а высоту ИО таблицы проводок=4. Количество строк зададим в блоке  $(1/\alpha)=30$ . Тогда:

$$t_{ac} = \frac{(2[\alpha v_j t] + 1)t_{ac.проводок} + 2t_{ac.поиска}}{(2[\alpha v_j t] + 3 + bl_{branch} + bl_{index})} =$$

$$\frac{(2[\alpha v_j t] + 1)t_{ac.проводок} + 2t_{ac.поиска}}{(2[\alpha v_j t] + 3 + 2h_{index} + 2(h_{branch} - 1))}$$

$$t_{ac} = \frac{(2[\alpha v_j t] + 1)t_{ac.проводок} + 2t_{ac.поиска}}{(2[\alpha v_j t] + 15)}$$

Ниже приведём на графиках полученные результаты, и оценим погрешность, вносимую фактом пренебрежения изменениями в трафике промежуточных блоков:

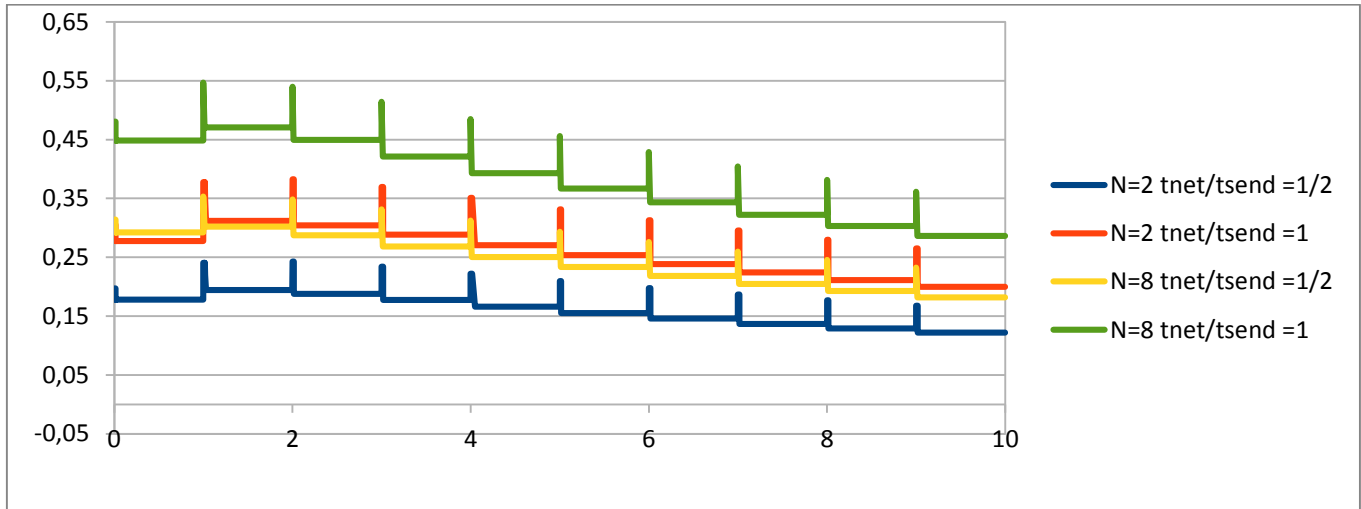


Рис. 4. Итоговый график 4-х подтрафиков

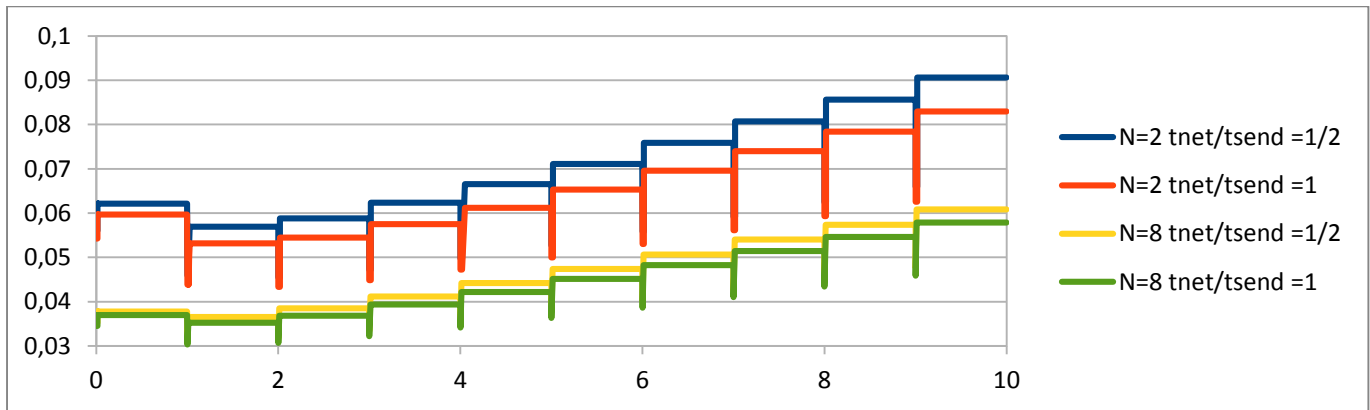


Рис. 5. График погрешности, вносимой промежуточными блоками при  $1/\alpha=30$

Из графика видно, что при 30 строках в блоке пренебрежение подтрафиком промежуточных блоков ИОТ допустимо, поскольку задержки, вносимые эти трафиком на порядок меньше времени рассчитанного отклика.

### Заключение

В работе было дано описание и математическое моделирование тестового трафика на основе проводок, названного простейшим, а также показана применимость описанной ранее математической модели [1] для безконфликтного простейшего трафика. Также много внимания было уделено разбивке трафиков на подтрафики для успешного моделирования. В ходе работы можно прийти к выводу, что критерием

эффективности разбивки трафика на подтрафики является полное (или преимущественное) непересечение подтрафиков по общим ресурсам. В противном случае совмещение подтрафиков порождает кардинально другой трафик, не связанный с начальным неразбитым трафиком. Дальнейшим направлением исследования будет применение найденной модели на простейшем трафике с конфликтами, применение модели [1] для ТРС-С трафика, а также оценка эффекта оптимизации кластерной системы от применения неблокирующего алгоритма фиксации распределённых транзакций [3] и неблокирующего алгоритма синхронизации облачных ресурсов.

### Список литературы

1. Гусев Е.И. Математическое моделирование распределённой кластерной системы использующей shared everything подход (Oracle RAC). – Вісник НТУУ "КПІ".Сер. Інформатика, управління та обчислювальна техніка. – 2014. – Випуск 60. – С.106-113.
2. Гусев Е.И. Исследование области применения неблокирующего алгоритма фиксации распределённых транзакций. – Вісник НТУУ "КПІ".Сер. Інформатика, управління та обчислювальна техніка. – 2012. – Випуск 57. – С.76-80.
3. Гусев Е.И., Кулаков А.Ю. Исключение блокирования общего ресурса в распределённых системах. – Вісник НТУУ "КПІ".Сер. Інформатика, управління та обчислювальна техніка. – 2011. – Випуск 54. – С.162-166.
4. Oracle® Real Application Clusters Administration and Deployment Guide 11g Release 2 (11.2) E16795-08
5. TPC BENCHMARK C Standard Specification Revision 5.11 , February 2010, Transaction Processing Performance Council (TPC), [www.tpc.org](http://www.tpc.org), © 2010 Transaction Processing Performance Council
6. Oracle® Database Administrator's Guide 12c Release 1 (12.1) E41484-10 August 2014. Primary Author: Randy Urbano Contributing Author: Leo Cloutier, Steve Fogel, Pradeep Gopal, Janis Greenberg
7. Oracle® Database New Features Guide 12c Release 1 (12.1) E49322-10 March 2015
8. Raman, Attaluri, Barber, Chainani, et al. (August 2013) "DB2 with BLU Acceleration: So Much More than Just a Column Store", Proceedings of the VLDB Endowment, Volume 6 Issue 11, Pages 1080-1091. Retrieved on February 1, 2014.