Eindhoven University
of Technology

Department of Mathematics and Computing Science

Memorandum COSOR 96-17

The Response Time Distribution in a Real-Time Database with Optimistic Concurrency Control

S.A.E. Sassen
J. van der Wal

Eindhoven, April 1996
The Netherlands

The Response Time Distribution in a Real-Time Database with Optimistic Concurrency Control*

S.A.E. Sassen and J. van der Wal
Department of Mathematics and Computing Science
Eindhoven University of Technology
P.O. Box 513, NL-5600 MB Eindhoven
The Netherlands

Abstract

For a real-time shared-memory database with optimistic concurrency control, an approximation for the transaction response time distribution is obtained. The model assumes that transactions arrive at the database according to a Poisson process, that every transaction uses an equal number of data-items uniformly chosen, and that the multiprogramming level is bounded. The analysis is based on a decomposition approach: results for the closed system with a fixed number of transactions are used to derive the response time distribution in the open system with Poisson arrivals. Numerical experiments that compare the analysis with simulation indicate that the approximation for the response time distribution works well for the closed system. For the open system the approximation is useful if the load is not too high.

*This research is supported by the Technology Foundation (STW), project EIF33.3129

The Response Time Distribution in a Real-Time Database with Optimistic Concurrency Control

Abstract

For a real-time shared-memory database with optimistic concurrency control, an approximation for the transaction response time distribution is obtained. The model assumes that transactions arrive at the database according to a Poisson process, that every transaction uses an equal number of data-items uniformly chosen, and that the multiprogramming level is bounded. The analysis is based on a decomposition approach: results for the closed system with a fixed number of transactions are used to derive the response time distribution in the open system with Poisson arrivals. Numerical experiments that compare the analysis with simulation indicate that the approximation for the response time distribution works well for the closed system. For the open system the approximation is useful if the load is not too high.

1 Introduction

Real-time databases combine the requirements of both databases and real-time systems. In a database, transactions (database requests) should preserve database consistency. Subject to this consistency requirement, the transaction throughput of the database should be maximised. In a real-time system, the main requirement is timeliness, i.e., transactions must be executed before their deadlines. Soft real-time systems are allowed to miss some deadlines when the system is overloaded, but the number of transactions that meets its deadline should be maximised. In a real-time database, both consistency and timeliness are important. We investigate soft real-time databases in this paper and are interested in the probability that a transaction meets its deadline.

To benefit from the increase in CPU power that parallel computer architectures offer, transactions on databases should be executed concurrently. However, concurrent execution can destroy database consistency if conflicting transactions are incorrectly scheduled. Two transactions conflict if they access the same data-item, at least one of them with the intention to write. Concurrency control schemes govern the simultaneous execution of transactions such that overall correctness of the database is maintained (see e.g. Papadimitriou [1986]). The two main concurrency control schemes are locking and optimistic concurrency control.

Under the locking scheme, an executing transaction holds locks on all data-items it needs for execution, thus introducing lock waits for transactions that conflict with it. Consistency is guaranteed, however chains of lock waits can lead to high transaction response times.

When the conflict probability is low, it can be advantageous to use the optimistic concurrency control (OCC) scheme proposed by Kung and Robinson [1981]. Under OCC, all CPUs can be used for transaction processing at the same time. Each transaction is processed in three phases: an execution phase, a validation phase and a commit phase. In the execution phase a transaction T accesses any data-item it needs for execution, regardless of the number of transactions already using that data-item. In the validation phase, all items used by T are checked for conflicts. If a conflict has occurred with a transaction that committed after T started, T must be rerun. If no conflicts occurred, T completes successfully and enters the commit phase, where the data-items used by T are updated.

There have been numerous performance studies of locking and optimistic concurrency control: simulation studies as well as analytical models. The only simulation study we mention here is by Agrawal et al. [1987], who give an extensive treatment of the influence the modelling assumptions with respect to resources and transaction behaviour can have on the outcome of a performance study. A collection of analytical models for locking is by Tay [1987]. Analytical models for OCC are (among others) by Menascé and Nakanishi [1982], and Kleinrock and Mehović [1992]. Actually, Menascé and Nakanishi compare the performance of locking and OCC. Other analytical comparative studies are by Morris and Wong [1985], and Yu et al. [1993].

All analytical studies cited above are stochastic analyses that only consider *average* system performance. Typical performance measures studied are the throughput, the mean response time, the probability of a lock wait, and the average number of restarts needed for a transaction under OCC. To our knowledge, except for the study of Bodlaender et al. [1996] for a variant of locking, no analytical performance studies of real-time databases exist that address the variance or the *distribution* of the response time. An approximation for the response time distribution is needed to estimate the probability that a transaction meets its deadline, or the percentage of transactions that missed its deadline; these quantities cannot be derived from the throughput alone.

In this paper we approximate the response time distribution in a multi-processor shared-memory database with optimistic concurrency control. The approach is based on the throughput analysis of Morris and Wong [1985]. The model is explained in section 2. In section 3, an approximation is derived for the response time distribution in a closed system with a fixed number of transactions, and its results are compared to simulation results. In section 4 we analyse the response time distribution in an open system with Poisson arrivals and test our approximation against simulation. Section 5 contains some concluding remarks and directions for future research.

2 The Model

We model OCC in a shared-memory environment with N parallel CPUs as a multi-server queueing system with feedback, see Figure 1 for an illustration.

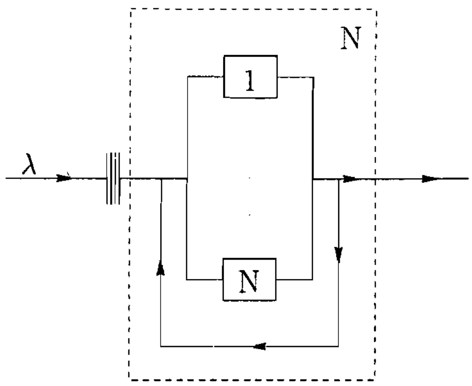


Figure 1: QUEUEING MODEL OF THE OPEN SYSTEM

In the dashed area, which represents the N CPUs, at most N transactions can be present. Each transaction is handled by one CPU and either leaves the system after a successful execution, or is rerun. We assume the time needed for one execution of a transaction is exponentially distributed with parameter μ. Further, it is assumed that the validation and commit phase take negligible time compared to the execution, and that the validation can be efficiently implemented such that parallel validation is possible. The assumption that validation and commit take negligible time compared to execution is reasonable, since there are no disks attached to the system and all data-items are in main memory. Transactions arrive at the database according to a Poisson process with rate λ. An arriving transaction that finds all CPUs busy joins the queue. As soon as a CPU is freed by a leaving transaction, the transaction that is first in queue is taken into execution. We also refer to execution plus validation as one transaction run.

The model described is largely the same as the model of Morris and Wong [1985]. With regard to transaction behaviour, in accordance with Morris and Wong we assume that

- each transaction uses an equal number a of data-items, uniformly picked from the total set d of data-items,
- the time needed for a rerun of a transaction exactly equals the time of the first transaction run, and
- all transactions write the data-items accessed, so there are no read-only transactions.

In light of these assumptions, we define b as the probability that two arbitrary transactions conflict. Then

$$b = 1 - \binom{d-a}{a} \Big/ \binom{d}{a}.$$

The queueing model of Figure 1 is no standard feedback model. The probability that a transaction T must be rerun is not fixed, but depends on the number of transactions that departed (committed) during the execution of T. The number of departures during T's execution depends on the length of T's execution and on the number of concurrently executing transactions.

We analyse the model using a decomposition approach. First we approximate the response time distribution in a so-called closed system with a constant population (multiprogramming level) of k transactions ($k \leq N$). The population is kept constant at k by admitting a new transaction to the system as soon as another transaction has committed. Next we consider the open system with Poisson arrivals and approximate the distribution of the response time using the results of the closed system. Such a decomposition approach has become fairly standard for computing the throughput and average response time of complicated queueing systems. We will show that in the model of Figure 1 the decomposition approach can also be applied for approximating the distribution of the response time.

For both the closed and the open system, numerical results from the analysis are compared with simulation results. In the simulation programs, the probability that a committing transaction conflicts with a transaction in execution is taken equal to b, in agreement with the above model description. Thus, no actual lists of data-items are used in the simulations. Also, the simulations take the time needed for a rerun of a transaction exactly equal to the time of the first transaction run.

3 Response Time Distribution in the Closed System

We consider the closed system depicted in Figure 2, for a fixed multiprogramming level k. After an execution run, a transaction can either be fed back for a rerun (requiring the same amount of time as the previous run), or leave. The moment a transaction leaves, a new transaction with a freshly drawn execution time enters the system.

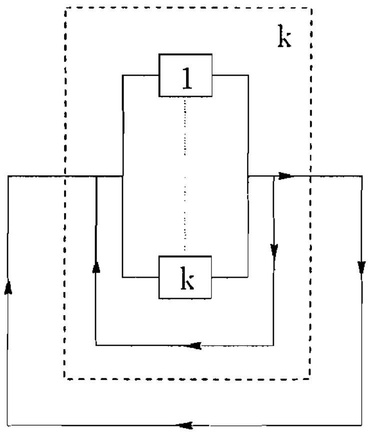


Figure 2: Queueing Model of the Closed System

3.1 Analysis

To analyse the response time distribution in this model, we use the idea of Morris and Wong [1985] and make the following assumption.

Commit Assumption

> A transaction T in execution observes other transactions to commit according to a Poisson process with rate α_k.

In order to determine the rate α_k at which T observes the other $(k-1)$ transactions to commit, we note that α_k is $(k-1)$ times the rate at which one CPU commits transactions. The total time a transaction spends in the system is ix if it requires i runs and has execution time x. Define $R_k(x)$ as the number of runs needed for a transaction with execution time x. Denote the response time of a transaction with execution time x by $S_k(x)$. Then $S_k(x) = R_k(x)x$. By the commit assumption, the probability that a transaction with execution time x has to be rerun is $1 - e^{-\alpha_k bx}$. Thus

$$P(R_k(x) = i) = \left(1 - e^{-\alpha_k bx}\right)^{i-1} e^{-\alpha_k bx}.$$

Hence, the expected response time of a transaction with execution time x is

$$E[S_k(x)] = \sum_{i=1}^{\infty} ixP(R_k(x) = i) = xe^{\alpha_k bx}.$$

Define S_k as the response time of a transaction in the closed system with population k. Conditioning on the execution time gives

$$E[S_k] = \int_0^\infty x e^{\alpha_k b x} \mu e^{-\mu x} \mathrm{d}x = \frac{\mu}{(\mu - \alpha_k b)^2}.$$

The long-term rate at which a CPU commits transactions is $1/E[S_k]$, so

$$\alpha_k = (k-1)\frac{(\mu - \alpha_k b)^2}{\mu}. \tag{1}$$

Now α_k can be solved from the quadratic equation (1). This yields

$$\alpha_k = \frac{\mu(1 + 2(k-1)b) - \mu\sqrt{1 + 4(k-1)b}}{2(k-1)b^2}, \tag{2}$$

the second root of (1) being excluded by the requirement $\mu - \alpha_k b > 0$, which is needed to ensure that $E[S_k]$ is finite. An interpretation of the requirement $\mu - \alpha_k b > 0$ is the following. Consider an arbitrary transaction T and define a T-invalidation epoch to be an epoch at which a transaction commits that conflicts with T. Then $\mu - \alpha_k b > 0$ says, that the average time needed for one run of T should not exceed the average time between two T-invalidation epochs.

In addition to the average response time $E[S_k]$, it is also possible to derive the distribution function of the response time S_k under the commit assumption. Consider a transaction T with execution time x. Then for $x \leq t$

$$\begin{aligned}
P(S_k(x) \leq t) &= \sum_{i=1}^{\infty} P(S_k(x) \leq t \mid R_k(x) = i) P(R_k(x) = i) \\
&= \sum_{i=1}^{\infty} 1_{\{ix \leq t\}} \left(1 - e^{-\alpha_k b x}\right)^{i-1} e^{-\alpha_k b x} \\
&= \sum_{i=1}^{\lfloor \frac{t}{x} \rfloor} \left(1 - e^{-\alpha_k b x}\right)^{i-1} e^{-\alpha_k b x} \\
&= 1 - \left(1 - e^{-\alpha_k b x}\right)^{\lfloor \frac{t}{x} \rfloor}.
\end{aligned}$$

Hence, under the commit assumption,

$$\begin{aligned}
P(S_k \leq t) &= \int_0^t \left[1 - (1 - e^{-\alpha_k b x})^{\lfloor \frac{t}{x} \rfloor}\right] \mu e^{-\mu x} \mathrm{d}x \\
&= 1 - e^{-\mu t} - \int_0^t (1 - e^{-\alpha_k b x})^{\lfloor \frac{t}{x} \rfloor} \mu e^{-\mu x} \mathrm{d}x,
\end{aligned} \tag{3}$$

with α_k given by (2).

Two remarks must be made. First, for high values of the conflict probability b (say $b > 0.6$), it can happen that the approximation (2) for α_k returns a value smaller than $\frac{k-1}{k}\mu$. This would imply that the throughput of the system, estimated as $\frac{k}{k-1}\alpha_k$, is smaller than μ which is not correct. The closed system always contains at least one successful transaction so has a throughput of at least μ. Thus, the approximation for α_k must be corrected. It can be shown that this correction is needed when $b > (k - \sqrt{k})/(k - 1)$. In agreement with Morris and Wong we modify (2) to read

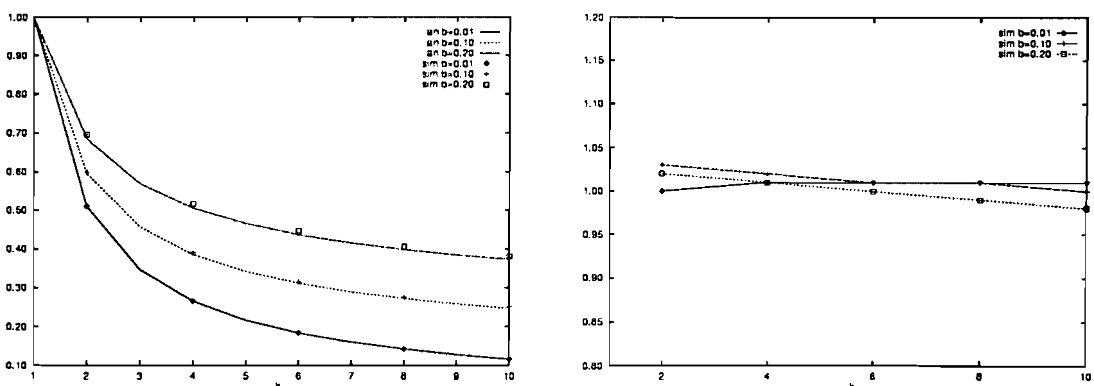
$$\alpha_k = \begin{cases} \dfrac{\mu(1 + 2(k-1)b) - \mu\sqrt{1 + 4(k-1)b}}{2(k-1)b^2} & \text{if } b \leq \dfrac{k - \sqrt{k}}{k - 1} \\ \dfrac{k-1}{k}\mu & \text{otherwise.} \end{cases}$$

Note that whenever this correction is necessary, OCC is not an attractive concurrency control algorithm anyway.

As a second remark, we note that it can be shown using (2), that α_k approaches μ/b as k approaches infinity. Also, α_k is strictly increasing in k. Hence, according to the analysis, the throughput $\frac{k}{k-1}\alpha_k$ of the closed system increases monotonically to μ/b. Thus if the number of CPUs becomes infinitely large, boundedness of the throughput implies unbounded response times (the requirement $\mu - \alpha_k b > 0$ is not fulfilled for k large).

3.2 Verifying the Commit Assumption

The quality of the approximation for the distribution of the response time depends on the quality of the commit assumption. Therefore, we studied the actual departure process by simulation. Let I_k be the interdeparture time, i.e., the time that elapses between two consecutive commits. According to the commit assumption, I_k is exponentially distributed with parameter $\frac{k}{k-1}\alpha_k$. In Figure 3(a) we plotted the expected interdeparture time $E[I_k]$ for systems with



(a) $E[I_k]$ (b) c_{I_k}

Figure 3: $E[I_k]$ AND c_{I_k} IN CLOSED SYSTEM

$\mu = 1$ and $k = 2$ to 10. Three cases of the conflict probability were considered: $b = 0.01$, $b = 0.1$, and $b = 0.2$. The lines show the values of $E[I_k]$ that follow from the analysis, and the symbols show the simulation results for $E[I_k]$. Figure 3(b) plots the simulated values of c_{I_k}, the coefficient of variation[1] of I_k. By the commit assumption, the interdeparture time

[1]The coefficient of variation c_X of a random variable X is defined as

$$c_X = \sqrt{Var(X)/E^2[X]}.$$

is assumed to be exponentially distributed so c_{I_k} equals 1. The simulation results for c_{I_k} are very close to 1. Note that the simulated values in Figure 3 are the midpoints of 95% confidence intervals and are confident up to 2 digits. It is clear from both Figure 3(a) and 3(b) that the commit assumption gives a good approximation to the departure process.

3.3 Numerical Results for the Response Time Distribution

Since the commit assumption is reasonable, we may expect that the approximation (3) of the response time distribution is quite good. We are interested in the probability $P(S_k > t)$ that a transaction does not meet its deadline t. For good system performance, this probability must be small. In this section, we test the quality of our approximation for $P(S_k > t)$. The value of $P(S_k > t)$ produced by the analysis is compared with the simulation result, for various t and k.

First, we used the analysis to find the values of t for which $P(S_k > t)$ equals 0.10, 0.05, and 0.01. These values of t are denoted by t_1, t_2, and t_3, respectively. The numerical integration method used for evaluating (3) was adaptive Simpson's quadrature (with an error smaller than 1E-5). Next, simulations were run to estimate $P(S_k > t_1)$, $P(S_k > t_2)$, and $P(S_k > t_3)$.

	μ_k			$P(S > t_1)$			$P(S > t_2)$			$P(S > t_3)$	
k	Sim	Ana	t_1	Sim	Ana	t_2	Sim	Ana	t_3	Sim	Ana
2	2.0	2.0	2.3	0.10	0.10	3.0	0.053	0.053	4.8	0.011	0.011
4	3.8	3.8	2.4	0.10	0.10	3.2	0.051	0.051	5.4	0.011	0.011
6	5.5	5.5	2.5	0.10	0.098	3.3	0.052	0.053	5.9	0.012	0.012
8	7.0	7.0	2.5	0.10	0.10	3.5	0.052	0.051	6.4	0.012	0.012
10	8.5	8.5	2.5	0.10	0.11	3.6	0.052	0.052	7.0	0.012	0.012

(a) $b = 0.01$

	μ_k			$P(S > t_1)$			$P(S > t_2)$			$P(S > t_3)$	
k	Sim	Ana	t_1	Sim	Ana	t_2	Sim	Ana	t_3	Sim	Ana
2	1.7	1.7	2.6	0.10	0.10	3.7	0.051	0.052	7.3	0.013	0.012
4	2.6	2.6	3.3	0.097	0.10	5.2	0.051	0.052	13	0.012	0.011
6	3.2	3.2	3.7	0.10	0.11	6.2	0.054	0.055	18	0.011	0.011
8	3.6	3.7	4.2	0.10	0.10	7.3	0.053	0.054	23	0.011	0.010
10	4.0	4.0	4.6	0.10	0.10	8.3	0.052	0.053	27	0.011	0.011

(b) $b = 0.1$

	μ_k			$P(S > t_1)$			$P(S > t_2)$			$P(S > t_3)$	
k	Sim	Ana	t_1	Sim	Ana	t_2	Sim	Ana	t_3	Sim	Ana
2	1.4	1.5	2.9	0.096	0.10	4.5	0.048	0.051	10	0.013	0.012
4	1.9	2.0	3.9	0.096	0.11	7.5	0.045	0.047	21	0.012	0.010
6	2.2	2.3	4.8	0.094	0.10	9.0	0.049	0.051	31	0.011	0.0097
8	2.5	2.5	5.9	0.088	0.096	10.7	0.049	0.051	37	0.012	0.011
10	2.6	2.7	6.9	0.083	0.090	12.2	0.048	0.050	46	0.011	0.010

(c) $b = 0.2$

Table 1: RESPONSE TIME DISTRIBUTION IN CLOSED SYSTEM WITH $\mu = 1$

Table 1 contains the simulation and analysis results for varying k, with $\mu = 1$, $b = 0.01$, 0.1 and 0.2. In a database of size $d = 1000$, $b = 0.01$ corresponds to a transaction size $a = 3$,

$b = 0.1$ to $a = 10$, and $b = 0.2$ to $a = 15$. Besides $P(S_k > t)$, the table also contains simulation and analysis results for the throughput of the closed system, denoted by μ_k. The simulated values in Table 1 are the midpoints of 95% confidence intervals and are confident up to 2 digits. Indeed, our analysis of the response time distribution closely matches the simulation. Only for $b = 0.2$, differences up to 10% can occur.

4 Response Time Distribution in the Open System

We now analyse the open system of Figure 1. Transactions arrive at the system according to a Poisson process with rate λ and transactions that find all CPUs occupied wait in a queue. The response time consists of waiting time plus the time spent executing inside the dashed area.

4.1 Analysis

The system of Figure 1 resembles an $M/G/N$ queueing system, with the exception that the service times are neither independent, nor identically distributed. As seen in section 3, the distribution of the response time of a transaction in the dashed area depends on the number of transactions concurrently in execution. In the open system, the number of transactions concurrently in execution changes through time. As no exact expression exists for the response time distribution of an $M/G/N$ system, there is not much hope for an exact expression for the response time distribution in the even more complicated model of Figure 1.

Therefore, an approximation for the response time distribution is desired. Define the throughput of a closed system with k customers by μ_k ($= \frac{k}{k-1}\alpha_k$ from section 3). The approximation we suggest consists of 3 parts:

1. Approximate the distribution of the waiting time W_i of a transaction that has i transactions waiting in front of it by an Erlang$(i+1, \mu_N)$ distribution.
2. Approximate the service time distribution of a transaction that finds $k-1$ CPUs busy on entering the database by $P(S_k \leq t)$, i.e., the response time of a transaction in a closed system with population k ($1 \leq k \leq N$). Approximate the service time distribution of a transaction that finds all CPUs busy on its arrival by $P(S_N \leq t)$.
3. Approximate the steady-state probabilities of having a total of i transactions in the system by

$$\pi_i = \begin{cases} \displaystyle\prod_{k=1}^{i} \left(\frac{\lambda}{\mu_k}\right) \pi_0 & , 1 \leq i \leq N \\ \left(\frac{\lambda}{\mu_N}\right)^{i-N} \pi_N & , i > N, \end{cases} \tag{4}$$

   where π_0 is computed from the normalisation condition $\sum_{i=0}^{\infty} \pi_i = 1$.

Support for part 1 is the fact that, as long as there are transactions waiting, the dashed area of Figure 1 behaves like a closed system with N transactions, for which the commit assumption gives a good approximation of the departure process. Thus the time until the next departure is approximately exponentially distributed with parameter μ_N.

Part 2 comes down to just ignoring that the number of concurrently executing transactions changes through time. Whether this yields a good approximation for the distribution of the time a transaction spends in the dashed area, cannot be said beforehand.

Approximation part 3 is (like 1) also based on the commit assumption. Under the commit assumption, the system behaves as a state-dependent exponential server with rate μ_k when k transactions are in service.

Using 1, 2, 3, and the PASTA property, we get as approximation for the distribution of the response time S in the open system:

$$P(S > t) = \sum_{i=0}^{N-1} \pi_i P(S_{i+1} > t) + \sum_{i=N}^{\infty} \pi_i P(W_{i-N} + S_N > t), \tag{5}$$

with W_i an Erlang$(i+1, \mu_N)$ distributed variable, and π_i given by (4).
Evaluating the second part of (5) by conditioning on W_i gives

$$\begin{aligned}
&\sum_{i=N}^{\infty} \pi_i P(W_{i-N} + S_N > t) \\
&= \sum_{i=N}^{\infty} \pi_N \left(\frac{\lambda}{\mu_N}\right)^{i-N} \int_0^{\infty} \frac{(\mu_N y)^{i-N}}{(i-N)!} \mu_N e^{-\mu_N y} P(S_N > t-y) \mathrm{d}y \\
&= \sum_{i=N}^{\infty} \pi_N \left(\frac{\lambda}{\mu_N}\right)^{i-N} \int_t^{\infty} \frac{(\mu_N y)^{i-N}}{(i-N)!} \mu_N e^{-\mu_N y} \mathrm{d}y \\
&+ \sum_{i=N}^{\infty} \pi_N \left(\frac{\lambda}{\mu_N}\right)^{i-N} \int_0^{t} \frac{(\mu_N y)^{i-N}}{(i-N)!} \mu_N e^{-\mu_N y} \left[\int_0^{t-y} (1 - e^{-\alpha_N b x})^{\lfloor \frac{t-y}{x} \rfloor} \mu e^{-\mu x} \mathrm{d}x + e^{-\mu(t-y)}\right] \mathrm{d}y,
\end{aligned}$$

which yields after some algebra

$$\begin{aligned}
\sum_{i=N}^{\infty} \pi_i P(W_{i-N} + S_N > t) &= \frac{\pi_N \mu_N}{\mu_N - \lambda} e^{-\mu_N t} e^{\lambda t} + \frac{\pi_N \mu_N}{\mu_N - \mu - \lambda}\left[e^{-\mu t} - e^{-(\mu_N - \lambda)t}\right] \\
&+ \pi_N \mu_N \int_0^t e^{-(\mu_N - \lambda) y} \int_0^{t-y} (1 - e^{-\alpha_N b x})^{\lfloor \frac{t-y}{x} \rfloor} \mu e^{-\mu x} \mathrm{d}x \, \mathrm{d}y. \quad (6)
\end{aligned}$$

As is shown in the appendix, the latter double integral can be reduced to the single integral

$$\begin{aligned}
\frac{\pi_N \mu_N}{\mu_N - \lambda} \int_0^t \Bigg\{ &\frac{1 - [e^{(\mu_N - \lambda)x}(1 - e^{-\alpha_N b x})]^{\lfloor \frac{t}{x} \rfloor - 1}}{1 - e^{(\mu_N - \lambda)x}(1 - e^{-\alpha_N b x})} e^{-(\mu_N - \lambda)(t-x)}(e^{(\mu_N - \lambda)x} - 1)(1 - e^{-\alpha_N b x}) \\
&+ (1 - e^{-(\mu_N - \lambda)(t - \lfloor \frac{t}{x} \rfloor x)})(1 - e^{-\alpha_N b x})^{\lfloor \frac{t}{x} \rfloor} \Bigg\} \mu e^{-\mu x} \mathrm{d}x.
\end{aligned}$$

Summarising, the approximation for the response time distribution is

$$
\begin{aligned}
&P(S > t) \\
&= \sum_{i=0}^{N-1} \prod_{k=1}^{i} \left(\frac{\lambda}{\mu_k}\right) \pi_0 \left\{ \int_0^t (1 - e^{-\alpha_{i+1} bx})^{\lfloor \frac{t}{x} \rfloor} \mu e^{-\mu x} \mathrm{d}x + e^{-\mu t} \right\} \\
&\quad + \pi_N \mu_N \left\{ \frac{e^{-(\mu_N - \lambda)t}}{\mu_N - \lambda} + \frac{e^{-\mu t} - e^{-(\mu_N - \lambda)t}}{\mu_N - \mu - \lambda} \right\} \\
&\quad + \frac{\pi_N \mu_N}{\mu_N - \lambda} \int_0^t \left\{ \frac{1 - [e^{(\mu_N - \lambda)x}(1 - e^{-\alpha_N bx})]^{\lfloor \frac{t}{x} \rfloor - 1}}{1 - e^{(\mu_N - \lambda)x}(1 - e^{-\alpha_N bx})} e^{-(\mu_N - \lambda)(t-x)} (e^{(\mu_N - \lambda)x} - 1)(1 - e^{-\alpha_N bx}) \right. \\
&\qquad\qquad \left. + (1 - e^{-(\mu_N - \lambda)(t - \lfloor \frac{t}{x} \rfloor x)})(1 - e^{-\alpha_N bx})^{\lfloor \frac{t}{x} \rfloor} \right\} \mu e^{-\mu x} \mathrm{d}x.
\end{aligned}
$$

The remaining integrals in the expression have to be evaluated numerically. They are of the same type as the integral in the approximating expression for $P(S_k > t)$ in section 3.

4.2 Numerical Results for the Response Time Distribution

We tested the quality of the approximation for $P(S > t)$ against simulation, for the three cases $b = 0.01$, $b = 0.1$ and $b = 0.2$ ($\mu = 1$). In order to prevent that the number of transactions in the system explodes, the transaction arrival rate λ_N for a system with N CPUs should not exceed μ_N, the maximum throughput when all N CPUs are used. Therefore, define

$$\rho = \frac{\lambda_N}{\mu_N}$$

as a measure for the load (utilisation) of the system. We considered systems with $\rho = 0.5$, 0.7, 0.8, and 0.9, for which numerical results are given in Table 2, 3, 4, and 5, respectively. Arrival rate λ_N was computed from $\lambda_N = \rho\mu_N$. The value μ_N was obtained from the analysis of the closed system. As seen in section 3, the analytical approximation for μ_N is quite good.

Each table contains simulation and analysis results for $E[S]$, $P(S > 2)$, $P(S > 5)$, $P(S > 10)$, and $P(S > 20)$. As the mean work requirement of a transaction is $1/\mu = 1$, $P(S > t)$ corresponds to the probability that a transaction spends more than t times its average required execution time in the system. Note that we evaluated the integrals in the approximation for $P(S > t)$ by adaptive Simpson's quadrature, with an error smaller than 1E-5.

For $\rho = 0.5$ and $\rho = 0.7$, the approximation for the response time distribution is very good: only minor differences with the simulation results can be observed. However the approximation for $E[S]$ is less accurate, especially for small N and $b = 0.1$ or $b = 0.2$.

For $\rho = 0.8$, the approximation for $P(S > t)$ is still very good, for all three values of b. Although the tail probabilities $P(S > t)$ are satisfyingly approximated, again we see that, especially for $b \neq 0.01$ and N small, the approximation for the average response time is not good.

In systems with $\rho = 0.9$ and $b \neq 0.01$ our approximation for the response time is not suitable: the analysis differs too much from the simulation. The conclusion from the numerical experiments is, that we have a good approximation for the response time distribution of a transaction in open systems with a load up to 0.8.

The results of the decomposition approach for the utilisations 0.8 and 0.9 are not as good as one would like them to be. At first sight, this was surprising. For under a high load, the system resembles the closed system of section 3, for which our approximation of the response time distribution performed satisfactorily. But we should have noted that an error of 5 percent in the throughput μ_N of the closed system results in an error of 5 percent for the utilisation in the open system. Under high system loads, an error in the approximation for the utilisation of the open system has a tremendous effect on the approximation for the response time. An increase of the utilisation from 0.90 to 0.95 approximately doubles the response time. The analysis for the closed system slightly overestimates the throughput. Thus, the actual load of the system of Table 5 is not 0.90 but higher. This explains that the simulation produced much larger values for $E[S]$ and $P(S > t)$ than the analysis. The approximation for the response time is too optimistic and under high loads this becomes a serious problem.

So we will have to improve the approximation Morris and Wong suggested for α_k. We think of the following. Consider the case that one of the CPUs is working on a fairly large job. Then this CPU will also not be committing for a long time. The other CPUs are not disturbed by this CPU and thus have less conflicts than usual. Hence they circulate faster and slow down the transaction at the CPU we are looking at even more. This effect is not captured by the present approximation.

5 Concluding Remarks and Further Research

We analysed the response time distribution in a real-time shared-memory database with optimistic concurrency control. The analysis went beyond previous performance studies because not the average but the entire distribution of the transaction response time was considered. Knowledge about the response time distribution is essential for the design of real-time database systems. For a given database design, the response time distribution indicates which percentage of transactions will miss their deadline. The number of CPUs needed to guarantee that this percentage does not exceed some prespecified value can thus be obtained from an approximation for the response time distribution.

Approximations were derived for the response time distribution in a database with a fixed multiprogramming level (the closed system), and for the response time distribution in a database where transactions arrive according to a Poisson process (the open system). The approximation for the response time distribution in the closed system was within a few percent of a simulation of the closed system. For the open system with utilisations up to 80%, the approximation also performed satisfactorily compared to simulation. However for higher loads, a small error in the approximation of the throughput leads to a large error in the approximation of the response time distribution. Then the approximation for the response time in the open system is too optimistic, so cannot be used to give performance guarantees. In order to get a better approximation for the open system, it is important to improve the throughput approximation for the closed system. This is a subject for further research.

Other topics for future research include extending the model for transactions that access a variable number of data-items, and for non-exponential work requirements. Whether the commit assumption is still valid in these situations is questionable and should be investigated.

Further, an analytical comparison of the response time distribution in databases with locking (see Bodlaender et al. [1996]) and OCC will be made. Finally, analytical models for the response time distribution in databases with more advanced OCC-schedulers (see e.g. Lee and Son [1993]) are desired. However, our analysis of the simplest OCC-scheme can already serve as a lower bound on the performance of these better schedulers.

Acknowledgements

We thank Maarten Bodlaender and Ad Aerts for useful discussions and their suggestions on the manuscript. In particular, we thank Sindo Núñez Queija for his help on simplifying the double integral (6).

References

Agrawal, R., M.J. Carey, and M. Livny [1987], Concurrency control performance modeling: alternatives and implications, *ACM Transactions on Database Systems* **12**, 609–654.

Bodlaender, M.P., S.A.E. Sassen, P.D.V. van der Stok, and J. van der Wal [1996], The response time distribution in a multi-processor database with single queue static locking, *Proceedings Fourth Int. Workshop on Parallel and Distributed Real-Time Systems, April 1996, Honolulu, Hawaii* , to appear.

Kleinrock, L. and F. Mehović [1992], Poisson winner queues, *Performance Evaluation* **14**, 79–101.

Kung, H. and J. Robinson [1981], On optimistic methods for concurrency control, *ACM Transactions on Database Systems* **6**, 213–226.

Lee, J. and S.H. Son [1993], Using dynamic adjustment of serialization order for real-time database systems, *Proceedings 14th Real-Time Systems Symposium, Raleigh-Durham, December 1993* , 66–75.

Menascé, D.A. and T. Nakanishi [1982], Optimistic versus pessimistic concurrency control mechanisms in database management systems, *Information Systems* **7**, 13–27.

Morris, R.J.T. and W.S. Wong [1985], Performance analysis of locking and OCC algorithms, *Performance Evaluation* **5**, 105–118.

Papadimitriou, C. [1986], *The Theory of Database Concurrency Control*, Computer Science Press, Rockville, Md..

Tay, Y.C. [1987], *Locking Performance in Centralized Databases*, Academic Press, Inc., Orlando, Florida.

Yu, P.S., D.M. Dias, and S.S. Lavenberg [1993], On the analytical modeling of database concurrency control, *Journal of the ACM* **40**, 831–872.

Appendix

We show that the double integral (6) can be simplified to a single integral. Below we write λ instead of $\mu_N - \lambda$, and α instead of $\alpha_N b$. By defining

$$I := \int_0^t \lambda e^{-\lambda y} \int_0^{t-y} (1-e^{-\alpha x})^{\lfloor \frac{t-y}{x} \rfloor} \mu e^{-\mu x} \mathrm{d}x \, \mathrm{d}y,$$

the integral (6) equals $(\pi_N \mu_N / \lambda) I$. Next, we concentrate on simplifying I.
Interchanging integrals and substituting $z = t - y$ gives

$$\begin{aligned} I &= \int_0^t \int_0^{t-x} -\lambda e^{-\lambda y} (1-e^{-\alpha x})^{\lfloor \frac{t-y}{x} \rfloor} \mu e^{-\mu x} \mathrm{d}y \, \mathrm{d}x \\ &= \int_0^t \int_x^t \lambda e^{-\lambda(t-z)} (1-e^{-\alpha x})^{\lfloor \frac{z}{x} \rfloor} \mu e^{-\mu x} \mathrm{d}z \, \mathrm{d}x. \end{aligned}$$

Now,

$$\begin{aligned} &\int_x^t \lambda e^{-\lambda(t-z)} (1-e^{-\alpha x})^{\lfloor \frac{z}{x} \rfloor} \mu e^{-\mu x} \mathrm{d}z \\ &= \sum_{j=1}^{\lfloor \frac{t}{x} \rfloor - 1} \int_{jx}^{(j+1)x} \lambda e^{-\lambda(t-z)} (1-e^{-\alpha x})^j \mathrm{d}z + \int_{\lfloor \frac{t}{x} \rfloor x}^t \lambda e^{-\lambda(t-z)} (1-e^{-\alpha x})^{\lfloor \frac{t}{x} \rfloor} \mathrm{d}z \\ &= \sum_{j=1}^{\lfloor \frac{t}{x} \rfloor - 1} (1-e^{-\alpha x})^j e^{-\lambda t} \left(e^{\lambda(j+1)x} - e^{\lambda j x} \right) + \left(e^{\lambda t} - e^{\lambda \lfloor \frac{t}{x} \rfloor x} \right) e^{-\lambda t} (1-e^{-\alpha x})^{\lfloor \frac{t}{x} \rfloor} \\ &= \sum_{j=1}^{\lfloor \frac{t}{x} \rfloor - 1} \left[e^{\lambda x} (1-e^{-\alpha x}) \right]^j e^{-\lambda t} \left(e^{\lambda x} - 1 \right) + \left(1 - e^{-\lambda(t - \lfloor \frac{t}{x} \rfloor x)} \right) (1-e^{-\alpha x})^{\lfloor \frac{t}{x} \rfloor} \\ &= \frac{1 - [e^{\lambda x}(1-e^{-\alpha x})]^{\lfloor \frac{t}{x} \rfloor - 1}}{1 - e^{\lambda x}(1-e^{-\alpha x})} e^{-\lambda(t-x)} (e^{\lambda x} - 1)(1-e^{-\alpha x}) + (1 - e^{-\lambda(t - \lfloor \frac{t}{x} \rfloor x)})(1-e^{-\alpha x})^{\lfloor \frac{t}{x} \rfloor}, \end{aligned}$$

so

$$\begin{aligned} I = \int_0^t \Bigg\{ &\frac{1 - [e^{\lambda x}(1-e^{-\alpha x})]^{\lfloor \frac{t}{x} \rfloor - 1}}{1 - e^{\lambda x}(1-e^{-\alpha x})} e^{-\lambda(t-x)} (e^{\lambda x} - 1)(1-e^{-\alpha x}) \\ &+ (1 - e^{-\lambda(t - \lfloor \frac{t}{x} \rfloor x)})(1-e^{-\alpha x})^{\lfloor \frac{t}{x} \rfloor} \Bigg\} \mu e^{-\mu x} \mathrm{d}x. \end{aligned}$$

	N	λ_N	μ_N	$E[S]$ Sim	$E[S]$ Ana	$P(S>2)$ Sim	$P(S>2)$ Ana	$P(S>5)$ Sim	$P(S>5)$ Ana	$P(S>10)$ Sim	$P(S>10)$ Ana	$P(S>20)$ Sim	$P(S>20)$ Ana
$b=0.01$	2	1.0	2.0	1.35	1.35	0.23	0.23	0.02	0.02	0.00	0.00	0.00	0.00
	4	1.9	3.8	1.12	1.12	0.16	0.16	0.01	0.01	0.00	0.00	0.00	0.00
	6	2.7	5.5	1.08	1.09	0.15	0.15	0.01	0.01	0.00	0.00	0.00	0.00
	8	3.5	7.0	1.08	1.09	0.15	0.15	0.02	0.02	0.00	0.00	0.00	0.00
	10	4.3	8.5	1.09	1.10	0.15	0.15	0.02	0.02	0.00	0.00	0.00	0.00
$b=0.10$	2	0.8	1.7	1.50	1.49	0.25	0.25	0.04	0.04	0.01	0.01	0.00	0.00
	4	1.3	2.6	1.38	1.38	0.19	0.19	0.04	0.04	0.01	0.01	0.00	0.00
	6	1.6	3.2	1.43	1.43	0.19	0.19	0.05	0.04	0.01	0.01	0.00	0.00
	8	1.8	3.7	1.51	1.49	0.19	0.19	0.05	0.05	0.02	0.02	0.00	0.00
	10	2.0	4.0	1.58	1.55	0.20	0.19	0.06	0.05	0.02	0.02	0.00	0.01
$b=0.20$	2	0.7	1.5	1.64	1.63	0.26	0.26	0.05	0.05	0.01	0.01	0.00	0.00
	4	1.0	2.0	1.60	1.58	0.21	0.21	0.06	0.05	0.02	0.02	0.00	0.01
	6	1.1	2.3	1.71	1.66	0.21	0.20	0.06	0.05	0.02	0.02	0.01	0.01
	8	1.3	2.5	1.81	1.74	0.21	0.20	0.07	0.06	0.03	0.02	0.01	0.01
	10	1.3	2.7	1.90	1.81	0.22	0.20	0.07	0.06	0.03	0.02	0.01	0.01

Table 2: RESPONSE TIME DISTRIBUTION IN OPEN SYSTEM FOR $\rho = 0.5$

	N	λ_N	μ_N	$E[S]$ Sim	$E[S]$ Ana	$P(S>2)$ Sim	$P(S>2)$ Ana	$P(S>5)$ Sim	$P(S>5)$ Ana	$P(S>10)$ Sim	$P(S>10)$ Ana	$P(S>20)$ Sim	$P(S>20)$ Ana
$b=0.01$	2	1.4	2.0	2.01	1.99	0.38	0.38	0.08	0.07	0.01	0.00	0.00	0.00
	4	2.6	3.8	1.42	1.42	0.24	0.24	0.02	0.02	0.00	0.00	0.00	0.00
	6	3.8	5.5	1.27	1.27	0.19	0.20	0.02	0.02	0.00	0.00	0.00	0.00
	8	4.9	7.0	1.21	1.22	0.18	0.18	0.02	0.02	0.00	0.00	0.00	0.00
	10	6.0	8.5	1.20	1.20	0.17	0.17	0.02	0.02	0.00	0.00	0.00	0.00
$b=0.10$	2	1.2	1.7	2.42	2.26	0.41	0.41	0.13	0.11	0.02	0.01	0.00	0.00
	4	1.8	2.6	1.91	1.87	0.29	0.30	0.08	0.07	0.02	0.02	0.00	0.00
	6	2.3	3.2	1.85	1.85	0.25	0.26	0.07	0.07	0.02	0.02	0.01	0.01
	8	2.6	3.7	1.90	1.91	0.23	0.24	0.07	0.07	0.03	0.02	0.01	0.01
	10	2.8	4.0	2.00	1.99	0.23	0.23	0.08	0.07	0.03	0.03	0.01	0.01
$b=0.20$	2	1.0	1.5	2.84	2.52	0.43	0.43	0.16	0.14	0.04	0.03	0.01	0.00
	4	1.4	2.0	2.37	2.26	0.31	0.32	0.11	0.10	0.03	0.03	0.01	0.01
	6	1.6	2.3	2.36	2.31	0.27	0.28	0.10	0.09	0.04	0.03	0.01	0.01
	8	1.8	2.5	2.51	2.42	0.25	0.26	0.10	0.09	0.04	0.04	0.02	0.01
	10	1.9	2.7	2.66	2.54	0.25	0.25	0.10	0.09	0.05	0.04	0.02	0.02

Table 3: RESPONSE TIME DISTRIBUTION IN OPEN SYSTEM FOR $\rho = 0.7$

	N	λ_N	μ_N	$E[S]$		$P(S>2)$		$P(S>5)$		$P(S>10)$		$P(S>20)$	
				Sim	Ana	Sim	Ana	Sim	Ana	Sim	Ana	Sim	Ana
$b=0.01$	2	1.6	2.0	2.88	2.83	0.51	0.51	0.17	0.17	0.03	0.02	0.00	0.00
	4	3.0	3.8	1.87	1.83	0.35	0.35	0.06	0.05	0.00	0.00	0.00	0.00
	6	4.4	5.5	1.56	1.55	0.27	0.27	0.03	0.03	0.00	0.00	0.00	0.00
	8	5.6	7.0	1.43	1.43	0.23	0.23	0.03	0.03	0.00	0.00	0.00	0.00
	10	6.8	8.5	1.37	1.37	0.21	0.21	0.03	0.03	0.00	0.00	0.00	0.00
$b=0.10$	2	1.3	1.7	3.65	3.24	0.55	0.54	0.24	0.21	0.07	0.05	0.01	0.00
	4	2.1	2.6	2.75	2.50	0.42	0.42	0.15	0.13	0.04	0.03	0.01	0.00
	6	2.6	3.2	2.50	2.36	0.35	0.36	0.12	0.10	0.04	0.03	0.01	0.01
	8	2.9	3.7	2.43	2.37	0.31	0.32	0.11	0.10	0.04	0.03	0.01	0.01
	10	3.2	4.0	2.44	2.43	0.28	0.30	0.10	0.10	0.04	0.04	0.01	0.01
$b=0.20$	2	1.2	1.5	4.68	3.66	0.58	0.56	0.30	0.25	0.12	0.07	0.03	0.01
	4	1.6	2.0	3.73	3.11	0.46	0.45	0.21	0.18	0.08	0.05	0.02	0.01
	6	1.8	2.3	3.39	3.06	0.38	0.39	0.17	0.15	0.06	0.05	0.02	0.02
	8	2.0	2.5	3.36	3.14	0.34	0.35	0.14	0.14	0.06	0.05	0.02	0.02
	10	2.1	2.7	3.43	3.27	0.31	0.33	0.13	0.13	0.06	0.05	0.02	0.02

Table 4: Response Time Distribution in Open System for $\rho = 0.8$

	N	λ_N	μ_N	$E[S]$		$P(S>2)$		$P(S>5)$		$P(S>10)$		$P(S>20)$	
				Sim	Ana	Sim	Ana	Sim	Ana	Sim	Ana	Sim	Ana
$b=0.01$	2	1.8	2.0	5.62	5.36	0.71	0.71	0.41	0.40	0.16	0.15	0.03	0.02
	4	3.4	3.8	3.31	3.13	0.57	0.56	0.21	0.20	0.04	0.03	0.00	0.00
	6	4.9	5.5	2.52	2.43	0.47	0.47	0.12	0.11	0.02	0.01	0.00	0.00
	8	6.3	7.0	2.21	2.10	0.41	0.40	0.09	0.08	0.01	0.01	0.00	0.00
	10	7.7	8.5	2.01	1.92	0.37	0.36	0.07	0.06	0.01	0.01	0.00	0.00
$b=0.10$	2	1.5	1.7	7.61	6.21	0.74	0.73	0.49	0.45	0.26	0.20	0.08	0.04
	4	2.3	2.6	6.23	4.41	0.66	0.63	0.40	0.32	0.20	0.10	0.06	0.01
	6	2.9	3.2	5.30	3.91	0.59	0.56	0.33	0.26	0.15	0.08	0.04	0.01
	8	3.3	3.7	4.80	3.73	0.54	0.52	0.28	0.22	0.13	0.07	0.03	0.01
	10	3.6	4.0	4.74	3.68	0.51	0.49	0.27	0.20	0.12	0.06	0.04	0.02
$b=0.20$	2	1.3	1.5	11.6	7.08	0.78	0.74	0.58	0.49	0.38	0.25	0.18	0.06
	4	1.8	2.0	11.5	5.63	0.71	0.66	0.51	0.39	0.33	0.17	0.16	0.03
	6	2.1	2.3	13.4	5.25	0.66	0.60	0.46	0.34	0.29	0.14	0.15	0.03
	8	2.3	2.5	8.29	5.17	0.59	0.56	0.39	0.30	0.23	0.12	0.10	0.03
	10	2.4	2.7	8.12	5.22	0.56	0.52	0.35	0.28	0.20	0.11	0.10	0.03

Table 5: Response Time Distribution in Open System for $\rho = 0.9$