Increasing the Expressiveness of Analytical Performance Models for Replicated Databases

Matthias Nicola Matthias Jarke

Technical University of Aachen, Informatik V (Information Systems)
Ahornstr. 55, 52056 Aachen, Germany
nicola@informatik.rwth-aachen.de

Abstract. The vast number of design options in replicated databases requires efficient analytical performance evaluations so that the considerable overhead of simulations or measurements can be focused on a few promising options. A review of existing analytical models in terms of their modeling assumptions, replication schemata considered, and network properties captured, shows that data replication and intersite communication as well as workload patterns should be modeled more accurately. Based on this analysis, we define a new modeling approach named 2RC (*2*-dimensional *r*eplication model with integrated *c*ommunication). We derive a complete analytical queueing model for 2RC and demonstrate that it is of higher expressiveness than existing models. 2RC also yields a novel bottleneck analysis and permits to evaluate the trade-off between throughput and availability.

1 Introduction

Replication management in distributed databases concerns the decision when and where to allocate physical copies of logical data fragments (replica placement), and when and how to update them to maintain an acceptable degree of mutual consistency (replica control). Replication intends to increase data availability in the presence of site or communication failures, and to decrease retrieval costs by local access if possible. The maintenance of replicated data is therefore closely related to intersite communication, and replication management has significant impact on the overall system performance.

The literature offers several algorithms for replica placement [32,22] as well as for replica control [8,9,15]. However, the sophistication of such methods is not matched by today's analytical models for performance evaluation. Considering the vast number of alternatives in the design of a replication schema, it becomes apparent that existing analytical models only consider very extreme replication schemata, e.g. no replication or full replication. Furthermore, the important role of intersite communication in replica management is not sufficiently taken into account. While the evolution and symbiosis of distributed database and modern communication systems is progressing, theoretical performance models to evaluate such systems lag behind. Our aim is to identify and remedy such flaws in the performance evaluations

Catriel Beeri, Peter Buneman (Eds.): ICDT'99, LNCS 1540, pp. 131-149, 1998.

and to derive an analytical modeling approach that describes real-world replicated databases more accurately so that new and more expressive results are obtained.

In section 2 we present a critical review of existing performance models including a formal classification of how replication can be modeled. Section 3 presents the requirements for a new comprehensive performance model (2RC) to overcome the deficiencies of existing approaches. A new 2-dimensional model of replication and the dependency structure captured in 2RC is described. Section 4 develops an analytical queueing model that implements 2RC. It is based on the 2D-replication model and a detailed communication model. Section 5 presents results derived from our model, including a bottleneck analysis which is the critical part of any analytical throughput estimation [33].

2 Shortcomings in Existing Performance Models

In this section we analyze alternatives in performance modeling of distributed databases which reveals drawbacks in existing studies and motivates 2RC.

2.1 General Modeling Concepts and Communication

Performance studies of distributed databases employed analytical methods [20,26,27, 16,13,2], as well as simulations [3,17,25,31,18,29,10,7]. Simulations can evaluate complex system models whose level of detail precludes analytical solutions. However, simulations are costly in terms of programming and computing time. Thus, simulations often fail to cover the parameter space and to carry out a sensitivity analysis as thoroughly as desired.

Simulation and analytical studies often use queueing systems as the underlying models. Early queueing models of distributed databases model a fully replicated database of *m* local sites by a M/M/m/FCFS system with blocking [4,11,24]. Read transactions are processed by the *m* servers in parallel, while write transactions block all *m* servers. This models shared reads and exclusive writes. Major drawbacks of these models are that intersite communication is neglected and all sites share a single queue of incoming transactions.

To remedy these flaws, distributed databases can be modeled by *queueing networks*. [10, 23] use networks of M/M/1 queues, i.e. each local database is modeled as an M/M/1 system. However, this still restricts all transactions to have the same exponentially distributed service times. More general, [5,16] model the local databases as M/G/1 queues with arbitrarily distributed service times, while [13,20] use networks of $M/H_2/1$ systems with 2-phase hyper-exponentially distributed service times to assign different exponentially distributed service times to read-only transactions and updates. Nevertheless, such models do not allow to evaluate real-world systems with more than two transaction types.

Most analytical performance studies model the communication network by an infinite server that introduces a constant delay for each message, regardless of

message size or network load [14,23,13,21,16,20]. "Infinite" means unlimited transmission capacity, no queueing of messages, and the network is never considered to be the bottleneck. Such models would predict that replication always deteriorates throughput but never increases it, which we will disprove in section 5. In modern, large wide area applications and in wireless and mobile information systems that suffer from low bandwidth, the communication links may indeed become a bottleneck. However, very few analytical studies tried to combine a detailed database model with a detailed communication model [2,26]. Most authors also assume uniformly distributed data access, i.e. each data item is accessed with equal probability [14,28,10,5,23,18, 29,21,20,3]. Non-uniform data access is more realistic but modeled in very few analytical studies [31,16]. Lock conflicts and blocking of transactions are usually only modeled to compare concurrency control algorithms [14,7,10,18,21]. Such models are of considerable complexity. They typically have to use simulations and simplified modeling assumptions concerning replication and communication.

2.2 Classification of Replication Models

Some performance studies of distributed databases simply assume *no replication*, i.e. each logical data item is represented by exactly one physical copy, [28,21]. Other models consider full or 1-dimensional partial replication:

(1) All objects to all sites (full replication)
Most performance evaluations assume full replication [11,14,4,24,23,18,29,27,3,17] i.e. all data objects are replicated to all sites so that each site holds a complete copy of the distributed database. This is an extreme case of replication and it has been recognized that for most applications neither full nor no replication is the optimal configuration [10,13,1].

(2) All objects to some sites (1-dimensional partial replication)
Several studies modeled partial replication in the way that each data object is replicated to some of the sites [7,10,16,31,20]. Formally, the degree of replication can be denoted by a parameter $r \in \{1,2,...,n\}$, describing that each logical data item is represented by r physical copies, where n is the number of sites. $r = 1$ expresses no replication, $r = n$ means full replication, and if $r > 1$, every data item is replicated. Consequently, either no or all data items are replicated. In many applications there is update-intensive data which should be replicated to very few sites while read intensive data should be replicated to many sites. This common situation cannot be modeled with the all-objects-to-some-sites scheme.

(3) Some objects to all sites (1-dimensional partial replication)
Alternatively, the degree of replication can be denoted by $r \in [0;1]$ describing the percentage of logical data items that are fully replicated to all sites. A data item is either fully replicated or not replicated at all. $r = 0$ expresses no replication, $r = 1$ means full replication. To the best of our knowledge, this model of partial replication has only been considered in [2,13,1]. The some-objects-to-all-sites scheme is orthogonal to the all-objects-to-some-sites approach since the degree of replication is

defined along the percentage of data items as opposed to the number of copies. However, in large wide area distributed databases it is hardly affordable to replicate some data items to all sites (causing considerable update propagation overhead) and others to none (reducing their availability severely). Thus, the some-objects-to-all-sites scheme is not realistic.

3 A 2D-Replication Model with Integrated Communication (2RC)

Learning from the existing models, we define the requirements for an improved modeling approach:

A more expressive model of replication is needed to represent and evaluate realistic replication schemata. Not only response time but also throughput and bottlenecks must be computable as important performance criteria. This requires to capture load dependent communication delay, network limited transaction throughput, and the interplay between replication and communication. Detailed transaction and communication patterns must be describable, so that real-world applications can be modeled. Non-uniform data access, the quality of replication schemata and relaxed coherency must also be considered.

All these requirements are met by 2RC. Additionally, the model of transaction processing in 2RC follows the primary copy approach [30], since it has been judged advantageous over other replica control concepts [15,18] and is implemented in commercial systems like Sybase and Oracle. In 2RC, asynchronous update propagation to the secondary copies is modeled, i.e. 2-phase-commit processing of updates is not considered, and transactions are assumed to be executed at a single site, either the local or a remote site. Since 2RC is not primarily intended to compare concurrency control algorithms, it refrains from modeling lock conflicts to allow for more details in the replication and communication submodels.

3.1 The 2-Dimensional Model of Replication

Based on the classification of section 2.2, the two orthogonal 1-dimensional concepts are combined to a new 2-dimensional scheme called "*Some objects to some sites*". In this, replication is modeled by a pair $(r_1,r_2) \in [0;1] \times \{2,...,n\}$ such that $r_1 \in [0;1]$ describes the percentage of logical data items which are represented by r_2 physical copies each, i.e. they are replicated to r_2 of the n sites. A share of $1 - r_1$ logical data items remain unreplicated. $r_1 = 0$ expresses no replication, ($r_1 = 1$, $r_2 = n$) means full replication.

For d logical data items, a replication schema (r_1,r_2) increases the number of physical copies from d (no replication) to $(r_1 \cdot d \cdot r_2) + (d \cdot (1 - r_1))$. Viewing the number of copies of replicated objects $(r_1 \cdot d \cdot r_2)$ as the actual extent of replication, we express it (for visualization and further calculations) independently from d and normalized to the interval [0;1] as an overall level of replication, yielding $(r_1 \cdot r_2)/n$. This characteristic of the 2D-approach is depicted in Fig. 1 for $n = 50$ sites.

Performance models which assume full replication only consider the point (1,*n*) as the possible replication schema. The all-objects-to-some-sites scheme analyses replication along the bold line from point (1,50) to (1,0) only. The orthogonal some-objects-to-all-sites scheme studies replication along the line from (1,50) to (0,50) only. However, the 2D-scheme considers any point in Fig. 1 a possible replication schema so that real-world replication strategies can be captured more accurately. Thus, we believe that the 2D-model is a profitable contribution towards a better understanding of how replication affects distributed system performance.

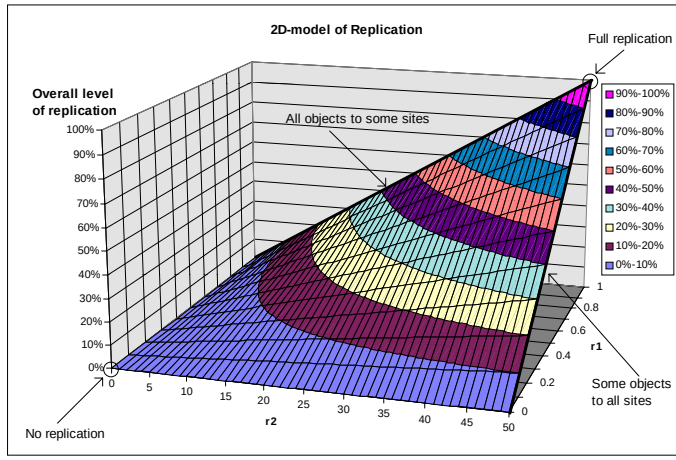

Fig. 1. The 2-dimensional model of replication

3.2 Dependency Structure

Apart from *how* the various aspects of a real system are modeled, it also matters which *dependencies* between them are considered in the model. Fig. 2 sketches the structure of dependencies we consider in 2RC. Rectangular nodes represent input parameters and modeling assumptions, oval nodes stand for intermediate or final results, the latter in bold. An arrow from a node A to a node B indicates that B depends directly on A. The 2D-replication scheme is a core part of our model and has direct impact on the quality of replication, the arrival rates and the network traffic, and thus substantial influence on all further results. τ transaction types (with different arrival rates and service time distributions) and 2 message types per transaction (with individually distributed transmission times depending on the message size) allow to model a wide range of different applications and workload patterns. The two bold arrows highlight the important dependencies through which load dependent communication delay and network limited throughput are captured. The overall throughput depends on both the network and the local database throughput, which allows a bottleneck analysis.

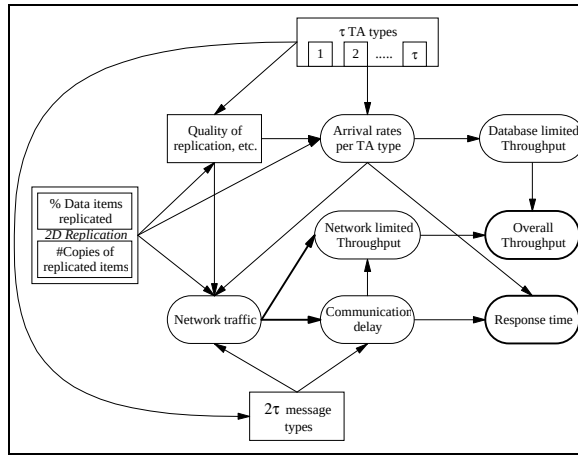

Fig. 2. Dependency structure of 2RC

For comparison, Fig. 3 shows the dependencies captured in [16,2] which we found typical for many analytical models. The model in [16] assumes constant communication delay and unlimited network capacity. On the database side, they consider 1D-replication and do not allow different types of queries or updates (like all models we examined). The model in [2] captures load dependent communication delay, but this is not exploited for throughput calculation and combined with a simple replication and transaction model. In [16] and [2] the arrival rates depend on non-uniform data access, modeled by a factor for access locality and hot spot access respectively, while such dependencies are neglected in most other studies. Almost all analytical studies fail to cover sufficient dependencies between transaction processing and communication, so that they do not calculate throughput as a performance criterion. In conclusion, 2RC combines a comprehensive database and replication model with a detailed communication model, and covers more interdependencies between the two than previous studies.

4 Analytical Queueing Model for 2RC

This section presents the mathematical implementation of a queueing model of a replicated database according to the 2RC approach. Table 1 shows the model parameters.

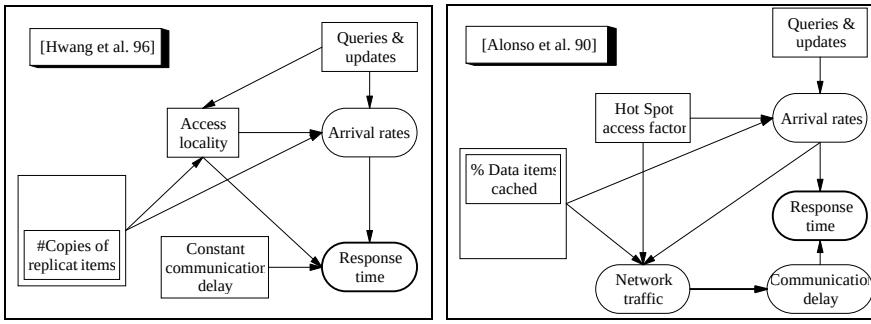

Fig. 3. Dependencies in Hwang et al. 96 [16] and Alonso et al. 90 [2].

Parameter	*Meaning*
n	Number of sites
τ	Number of transaction types
a_i	Percentage of transactions of type i
q_i	Function to distinguish between queries and updates
λ_i	Transaction arrival rate per site (TPS) of TA type i
r_1	Percentage of data items replicated
r_2	No. of copies of replicated data items
t_i	Mean service time for a transaction of type i (sec)
k	Coherency index
baud	Communication bandwidth (bps)
loc_i	Transactions' locality without replication
$plcmt_i$	Quality of replica placement
sel_i	Quality of replica selection
f_plcmt_i	Fine tuning replica placement
f_sel_i	Fine tuning replica selection
ℓ_i	Probability of local transaction execution
$size_c^{send\_i}$	Message size for a send of a transaction of type i.
$size_c^{return\_i}$	Message size of returned query results (byte)
$t_c^{send\_i}$	Mean time to send a transaction of type (sec)
$t_c^{return\_i}$	Mean time to return query results (sec)

Table 1. Model parameters.

4.1 Workload

We model a replicated database by an open queueing network in which each of the n identical sites is represented by a $M/H_\tau/1$ system. Transaction arrivals to the distributed system are modeled by n identical Poisson processes with parameter λ, i.e.

one arrival stream per site such that the transaction load increases with the number of sites. The τ different types of transactions are numbered 1,2,.....,τ. A transaction is with probability a_i of type i, $\sum_{i=1}^{\tau} a_i = 1$, so the Poisson arrival process at a site consists of τ separate streams having rate $\lambda_i = a_i \cdot \lambda$, $1 \le i \le \tau$. A characteristic function q distinguishes between queries and updates:

$$q\text{: } \{1,2,...,\tau\} \rightarrow \{0,1\} \text{ with } q(i) = q_i = \begin{cases} 1, \text{if transactions of type } i \text{ are queries} \\ 0, \text{if transactions of type } i \text{ are updates} \end{cases}$$

The number of data objects accessed per transaction is assumed to be geometrically distributed, so the service time for a transaction of type i at a local database is modeled as exponentially distributed with mean t_i (seconds) [28]. Thus, the service time for the combined arrival process of all τ transaction types follows a τ-phase hyperexponential distribution.

4.2 Locality and Quality of Replication

Without replication but due to skillful data allocation transactions exhibit a behaviour of locality, i.e. they tend to access data items locally available at their site of submission. This is modeled by the probability $loc_i \in [0;1]$ ($1 \le i \le \tau$) that a transaction of type i can be executed at the local site, while it has to be forwarded to a remote site with probability $1 - loc_i$. Introducing partial replication (r_1,r_2) then increases the probability that a query can be answered locally by $(r_1 \cdot r_2)/n$. Due to the primary copy approach, the write availability does not increase.

The *selection* of data items to replicate and the decision where to *place* them can have significant impact on the overall system performance [32]. Thus, we consider this quality of a replication design in the performance evaluation. We find that *replica selection* has major impact on updates, while *replica placement* is more significant for query processing:

Updates: Selecting many update intensive data items for replication causes high update propagation overhead. Therefore we introduce a "selection parameter" $sel_i \in [0;1/r_1]$ for $1 \le i \le \tau$ and $q(i) = 0$, expressing to which extent updates of type i tend to access data items that were selected for replication. $sel_i = 0$ means that the replicated data items are never changed by updates of type i. $sel_i = 1$ signals no preferences towards replicated or unreplicated data, and $sel_i = 1/r_1$ declares that replica selection is so unfortunate that updates of type i always access replicated data.

Queries: Even if all read intensive data items are replicated, performance gains are quite low as long as these replicas are not available locally to the queries. Thus, a "placement parameter" $plcmt_i \in [1; n/(r_1 \cdot r_2)]$ for $1 \le i \le \tau$ and $q(i) = 1$ expresses to which extent replica placement is increasing the probability that a query of type i can be executed locally. $plcmt_i = 1$ means that replica placement does not necessarily

increase the queries' locality, and $plcmt_i = n/(r_1 \cdot r_2)$ declares that queries of type i can always run at their home sites.

The higher the degree of replication the more likely it is that not only update intensive data items are selected for replication, and the more likely it is that replication increases the local read probability. Therefore, the parameters sel_i and $plcmt_i$ have to depend on the degree of replication and are defined as

$$plcmt_i = \frac{1}{\left(\frac{r_1 \cdot r_2}{n}\right)^{f\_plcmt_i}} \quad \text{and} \quad sel_i = \frac{1}{(r_1)^{f\_sel_i}}$$

so that the parameters $f\_plcmt_i \in [0,1]$ and $f\_sel_i \in [-\infty,1]$ can „*fine-tune*" how far or close $plcmt_i$ and sel_i follow their optimum values $n/(r_1 \cdot r_2)$ and $1/r_1$ respectively.

We assume that the replicas are distributed evenly across the sites, so that each site receives an equal share of forwarded transactions and propagated updates. Thus, the overall probability ℓ_i that a transaction of type i ($1 \le i \le \tau$) can be executed at its local site amounts to

$$\ell_i = loc_i + q_i \cdot (1 - loc_i) \cdot \frac{r_1 \cdot r_2}{n} \cdot plcmt_i$$

because without replication a transaction is processed locally with probability loc_i (first term). With probability $1 - loc_i$ queries ($q_i = 1$) cannot be executed locally, but replication increases the local read availability by $(r_1 \cdot r_2)/n$ or higher, depending on the replica placement (second term). Note, that loc_i, sel_i and $plcmt_i$ model non-uniform data access.

4.3 Transaction Processing and Arrival Rates

The performance of replicated databases can be improved if the requirement of mutual consistency among the replicas of a logical data item is relaxed. Various concepts of relaxed coherency can be denoted by coherency conditions which allow to calculate a coherency index $k \in [0;1]$ as a measure of the degree of allowed divergence [13]. Small values of k express high relaxation, $k = 0$ models suspended update propagation, and for $k = 1$ updates are propagated immediately.

For $1 \le i \le \tau$, the total arrival rate λ_i^{total} of transactions of type i at a single site amounts to

$$\lambda_i^{total} = \ell_i \cdot \lambda_i + (n-1) \cdot (1-\ell_i) \cdot \lambda_i \cdot \frac{1}{n-1} + (1-q_i) \cdot (n-1) \cdot (r_1 \cdot sel_i) \cdot \frac{r_2 - 1}{n-1} \cdot k \cdot \lambda_i$$

because a share of ℓ_i of the incoming λ_i transactions can be executed locally (first term) whereas the remaining $(1-\ell_i) \cdot \lambda_i$ transactions are forwarded to sites where appropriate data is available. The other n-1 sites also forward $1-\ell_i$ of their λ_i

transactions which are received by each of the remaining databases with equal probability 1/(*n*-1). This explains the second term. The third term covers update propagation and its possible reduction through relaxed coherency: If transactions of type *i* are updates (i.e. 1 - *q*(*i*) = 1) the local load λ_i^{total} is increased by update propagation from the *n*-1 remote sites. The probability that an update at one of the *n*-1 remote sites hits a primary copy which is replicated, is $r_1 \cdot sel_i$. The probability, that one of the corresponding secondary copies resides at the local database is (r_2-1)/(*n*-1) because the r_2 - 1 secondary copies are distributed evenly over *n*-1 sites. Finally, update propagation may be reduced by relaxed coherency, i.e. if $k < 1$. The above formula simplifies to

$$\lambda_i^{total} = \lambda_i + (1 - q_i) \cdot (r_1 \cdot sel_i) \cdot (r_2 - 1) \cdot k \cdot \lambda_i \quad \text{and we define} \quad \lambda^{total} := \sum_{i=1}^{\tau} \lambda_i^{total} .$$

4.4 Intersite Communication

Two messages are required to execute a transaction at a remote site: a *send* and a *return*, e.g. a query is sent to a site and the result is returned. For each transaction type *i* the communication delay for a *send* (*return*) is modeled to be exponentially distributed with mean $t_c^{send\_i}$ ($t_c^{return\_i}$) seconds ($1 \le i \le \tau$). These values mainly depend on the bandwidth and the message size. Therefore, the parameter *baud* represents the network's bandwidth in bps, and $size_c^{send\_i}$ ($size_c^{return\_i}$) denotes the message size in bytes for a *send* (*return*) of a transaction of type *i*. The means $t_c^{send\_i}$ and $t_c^{return\_i}$ characterizing the exponentially distributed service times are hence defined as

$$t_c^{send\_i} = \frac{8 \cdot size_c^{send\_i}}{baud} \quad \text{and} \quad t_c^{return\_i} = \frac{8 \cdot size_c^{return\_i}}{baud}$$

The average number of messages per second in the distributed system amounts to

$$\overline{N} = \sum_{i=1}^{\tau} n \cdot (1 - \ell_i) \cdot \lambda_i + (1 - q_i) \cdot n \cdot (r_1 \cdot sel_i) \cdot (r_2 - 1) \cdot k \cdot \lambda_i + \sum_{i=1}^{\tau} q_i \cdot n \cdot (1 - \ell_i) \cdot \lambda_i \qquad (*)$$

The first sum covers messages of type *send* (transactions forwarded to remote sites due to a lack of appropriate local data and update propagation), the second sum are returned query results. Remote updates are assumed not to be acknowledged and thus do not cause return messages. To model limited network capacity, the local databases are considered to be connected to the network via local communication servers modeled as $M/H_{2\tau}/1$ systems. The arrival rate at any such server is $\overline{N}/n$ messages per second because each site sends and receives the same number of messages due to the sites' identical layout and symmetrical behaviour. The service time follows an $H_{2\tau}$ distribution, because τ transaction types imply 2τ different message types: τ

message types have an exponentially distributed service time with mean $t_c^{send\_i}$, and τ message types with mean $t_c^{return\_i}$ $(1 \le i \le \tau.)$. The expression (*) implies, that a share of $(n \cdot (1-\ell_i) \cdot \lambda_i + (1-q_i) \cdot n \cdot (r_1 \cdot sel_i) \cdot (r_2 - 1) \cdot k \cdot \lambda_i) / \overline{N}$ of the $\overline{N}$ messages has a mean service time $t_c^{send\_i}$ (for $1 \le i \le \tau$) and a share of $(q_i \cdot n \cdot (1-\ell_i) \cdot \lambda_i) / \overline{N}$ of the messages has a mean service time $t_c^{return\_i}$ (for $1 \le i \le \tau$). Hence, the first and second moment of the $H_{2\tau}$ service time distribution can be derived following [19]. Subsequently, the average waiting time $\overline{W_c}$ at a local communication server can be obtained using the *Pollaczek-Khinchin formula* for general M/G/1 systems [19] and results in

$$\overline{W_c} = \frac{\sum_{i=1}^{\tau} \left((1-\ell_i) \cdot \lambda_i + (1-q_i) \cdot (r_1 \cdot sel_i) \cdot (r_2 - 1) \cdot k \cdot \lambda_i\right) \cdot \left(t_c^{send\_i}\right)^2 + q_i \cdot (1-\ell_i) \cdot \lambda_i (t_c^{return\_i})^2}{1 - \sum_{i=1}^{\tau} \left((1-\ell_i) \cdot \lambda_i + (1-q_i) \cdot (r_1 \cdot sel_i) \cdot (r_2 - 1) \cdot k \cdot \lambda_i\right) \cdot t_c^{send\_i} + q_i \cdot (1-\ell_i) \cdot \lambda_i \cdot t_c^{return\_i}}$$

4.5 Performance Criteria

Similar to the calculation of $\overline{W_c}$, the mean waiting time $\overline{W}$ at a local database is found to be

$$\overline{W} = \frac{\lambda^{total} \cdot \sum_{i=1}^{\tau} \frac{2 \cdot \lambda_i^{total}}{\lambda^{total}} \cdot t_i^{\,2}}{2 \cdot \left(1 - \lambda^{total} \cdot \sum_{i=1}^{\tau} \frac{\lambda_i^{total}}{\lambda^{total}} \cdot t_i\right)} = \frac{\sum_{i=1}^{\tau} \lambda_i^{total} \cdot t_i^{\,2}}{1 - \sum_{i=1}^{\tau} \lambda_i^{total} \cdot t_i}$$

so that the combined average response time over all transaction types results in

$$\overline{R} = \sum_{i=1}^{\tau} a_i \cdot \overline{R_i} \quad , \quad \text{where} \quad \overline{R_i} = \overline{W} + t_i + (1-\ell_i) \cdot (\overline{W_c} + t_c^{send\_i} + t_c^{return\_i})$$

is the response time for transactions of type *i*. On average a transaction (of type *i*) needs to wait for $\overline{W}$ seconds at a database to receive a service of t_i seconds. Additionally, with probability $(1-\ell_i)$ a transaction needs to be forwarded to a remote site which takes $\overline{W_c}$ seconds to wait for plus the time to be sent and returned. Note, th we assume $t_c^{return\_i} = 0$ if $q(i) = 0$, i.e. no return messages for updates.

In steady state, the throughput of the local databases equals the arrival rate λ but is bound by the limited system capacity. Specifically, the throughput can grow until either a local database server or a communication server (the network) is saturated, i.e. its utilization (ρ_D or ρ_C respectively) reaches 1. Solving the equations $\rho_D = 1$ and $\rho_C = 1$ for the arrival rate λ yields the maximum database limited throughput T_D and the maximum communication/network limited throughput T_C as

$$T_D = \lambda = \left(\sum_{i=1}^{\tau} a_i \cdot \left(1+(1-q_i)\cdot(r_1 \cdot sel_i)\cdot(r_2-1)\cdot k\right)\cdot t_i \right)^{-1} \quad \text{and}$$

$$T_C = \left(\sum_{i=1}^{\tau} \left((1-\ell_i)\cdot a_i + (1-q_i)\cdot(r_1 \cdot sel_i)\cdot(r_2-1)\cdot k \cdot a_i\right)\cdot t_c^{send\_i} + q_i \cdot (1-\ell_i)\cdot a_i \cdot t_c^{return\_i} \right)^{-1}$$

respectively. The maximum throughput at a database site is $T = min(T_D,T_C)$ because whatever server is saturated first (either the database or the communication server) is the bottleneck and limits throughput. The overall system throughput amounts to $n \cdot T$.

5 Results and Expressiveness of the 2RC-Approach

The results we present are based on parameter values which are carefully chosen after measurements in database systems for telecom applications [12]. The values also agree with [2,10]. For ease of presentation, we consider only 3 transaction types: short queries, updates, and long queries (e.g. statistical evaluations).

Parameter	*Base setting*
n	50
τ	3
q_i	$q_1 = 1$, $q_2 = 0$, $q_3 = 1$
a_i	$a_1 = 0.85$, $a_2 = 0.1$, $a_3 = 0.05$
loc_i	0.04
f_plcmt_i	0.55
f_sel_i	-0.8
k	1
t_i	$t_1 = 0.06$, $t_2 = 0.125$, $t_3 = 0.5$
baud	64000 (e.g. ISDN)
$size_c^{send\_i}$	400 byte each
$size_c^{return\_1}$	2000
$size_c^{return\_3}$	3500

The sensitivity analysis (omitted here for brevity) showed that parameter variations affect the performance values in a reasonable way, i.e. the model is stable. Although we will mention absolute values to refer to characteristics in the diagrams below, we consider the general trends, shapes and expressiveness of the graphs as the primary results. Naturally, the graphs depend on input values like the number of updates etc. However, here we do not intend to compare results for many different parameter combinations rather than to show that the 2RC approach allows a more expressive analysis for *any* set of parameters.

Fig. 4 shows the maximum *throughput* $T(r_1,r_2) = min(T_D,T_C)$ over the r_1-r_2-*space*. A 1D-replication model considers either the „$r_1 = 1$ -edge“ of the graph, or the „$r_2 = 50$ -edge“. Either case merely expresses that the throughput increases with a moderate degree of replication ($r_1 = 0.3$ or $r_2 = 10$) but decreases remarkably when replication is medium or high. (Note: Models which assume unlimited network capacity cannot foresee this increase, e.g. [13,16,30].) However, the 2D-model tells us more: As long as less than 35% of the data items are replicated ($r_1 < 0.35$) throughput can be maximized by placing copies on all sites ($r_2 = 50$), reaching its highest peak of 320 TPS for $r_1 = 0.3$. If availability considerations require more data items to be replicated (e.g. 50%), a medium number of copies yields the maximum throughput (e.g. $r_2 = 30$ for $r_1 = 0.5$). When striving for very high availability, it is

worthwhile to consider that replicating 75% of the data items to 40 sites allows a twice as high throughput as full replication. These results show the increased expressiveness of the 2D-model over 1-dimensional approaches.

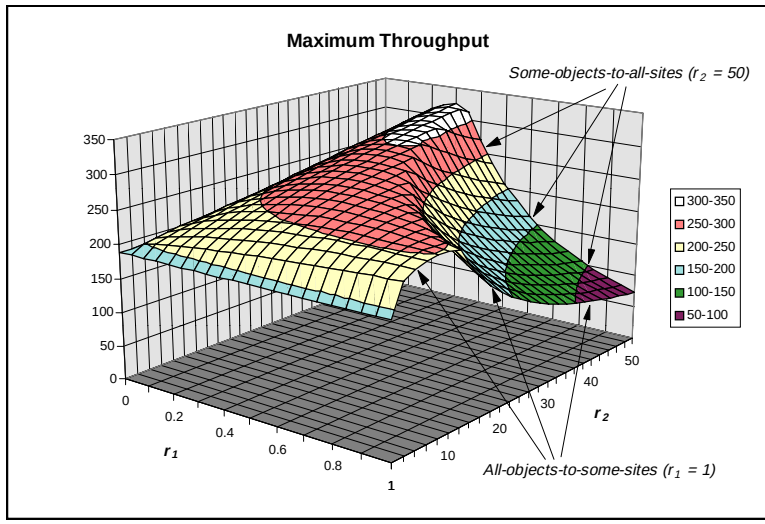

Fig. 4. Maximum Overall System throughput in TPS

The result that throughput can be maximized by placing copies at all sites for $r_1 < 0.35$ depends on the quality of replication. The input value $f\_sel_i = -0.8$ models a fairly good replication schema in which mainly read intensive but very little update intensive data items are selected for replication. With such a careful replica selection the update propagation overhead does not increase dramatically as the number of copies (r_2) is maximized to enhance local read availability. This relieves communication and increases the overall throughput. For larger values of r_1 ($r_1 \rightarrow 1$) it becomes increasingly difficult to avoid replication of update intensive data items such that placing copies at all sites is not affordable anymore (cf. Fig. 5). Consequently, Fig. 4 looks different for a *bad* replica selection ($f\_sel_i > 0$), i.e. the throughput peak is found at $r_1 = 1$ for a low number of copies ($r_2 \leq 12$). The more data items are replicated ($r_1 \rightarrow 1$) the more difficult it is to pick *only* update intensive objects for replication such that the read availability increases. This is captured in the model because sel_i converges towards 1 (i.e. unbiased replica selection) for $r_1 \rightarrow 1$ even for *bad* settings of f_sel_i.

The results presented throughout this section and the benefits of replication also depend on the percentage of updates in the overall workload. With an increasing percentage of updates the benefits of replication are gradually reduced and eventually outweighed by its drawbacks (update propagation cost vs. local and parallel read access) such that no replication ($r_1 = 0$ / $r_2 = 1$) yields the best performance.

Since the throughput calculation considered limited network capacity, a *bottleneck analysis* is possible for any replication schema and can be read off Fig. 5: For low replication (r_1 < 0.3 or r_2 < 10), remote data access saturates the network earlier than update propagation saturates the local databases. For medium and high replication (r_1 > 0.5, r_2 > 25) more update propagation network traffic is generated, but at the same time increased local data availability relieves the network, while the transaction load to update secondary copies grows and causes the local databases to be the throughput bottleneck.

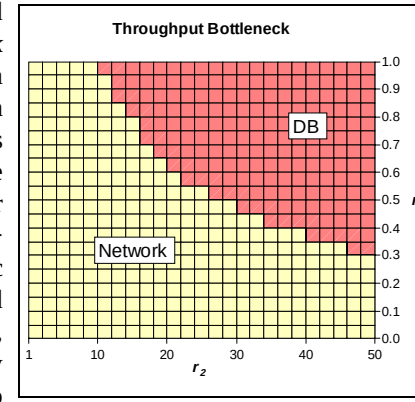

Fig. 5. Bottleneck Analysis

The combined average *response time* $\overline{R}$ is shown in Fig. 6. A moderate degree of replication (r_1 < 0.5, r_2 < 20) leads to increased local and parallel data access and thus reduces response time from over 1 to less than half a second. Extending replication along *both* dimensions rapidly saturates the system with propagated updates which outweigh the advantage of local read access and cause high response times. The 2D-model reveals that low response times are still possible if replication is extended along one dimension but kept moderate in the other. Replication schemata as different as (r_1 = 1, r_2 = 10), (r_1 = 0.3, r_2 = 50) or (r_1 = 0.5, r_2 = 30) could be chosen to satisfy an application`s individual requirements regarding reliability and availability, while retaining similarly low response times. 1-dimensional models of replication did not allow this kind of examinations.

To address *scalability* issues, Fig. 7 depicts the throughput as a function of r_1 and the system size *n*. Note that the model considers one transaction arrival stream per site such that the overall transaction load increases as the number of sites increases. Furthermore, the number of logical data objects increases with the number of sites. As an example, r_2 is set to 2*n*/3 for any value of *n*, i.e. r_1% of the logical data items are replicated to 2/3 of the *n* sites. If 100% of the data items are replicated, the throughput grows to about 100 TPS as the number of sites is increased to 30. Larger systems do not achieve a significantly higher throughput because without relaxed coherency high replication in large systems causes considerable update propagation overhead which hinders scalability [15]. Reducing replication (r_1 < 1) gradually improves scalability. If less than 40% of the data objects are replicated, far fewer update propagation transactions have to be processed so that the databases are not the bottleneck anymore (see Fig. 7) and throughput can grow to over 500 TPS for r_1 = 0.25.

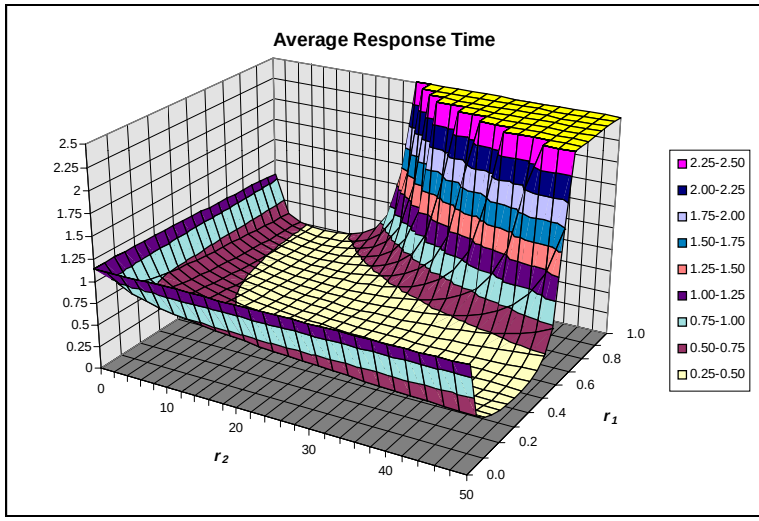

Fig. 6. Average overall response time $\overline{R}(r_1, r_2)$

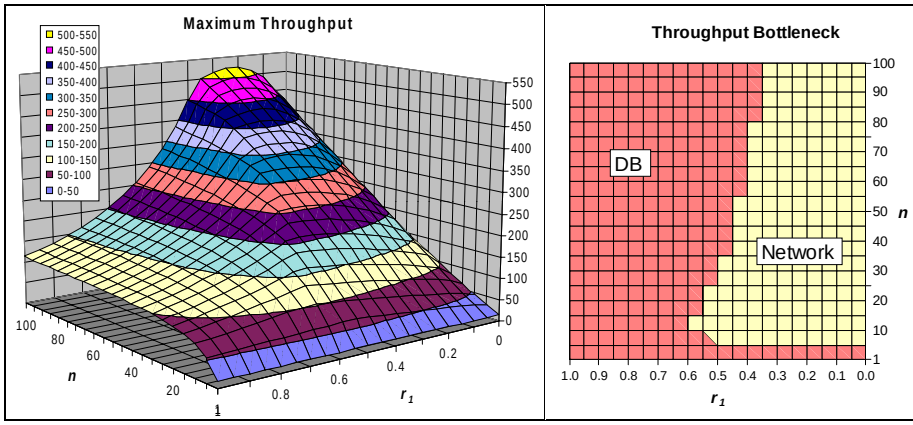

Fig. 7. Overall throughput $T(r_1,n)$ with Bottleneck Analysis

As indicated in the discussion for Fig. 4, the way throughput and scalability can be improved by means of replication depends critically on the selection of data items for replication and the placement of their copies. In Fig. 8 we examine the impact of the quality of replication on the scalability by considering a bad replication schema, i.e. one in which (a) not only read intensive but also a considerable amount of update intensive data items are replicated and (b) the copies are placed at randomly chosen sites rather than at remote sites where they are read particularly often. This can be modeled by the parameter settings $f\_plcmt_i = 0$ and $f\_sel_i = 0.6$. Such a replication schema causes more replica maintenance overhead than benefits through local and parallel read access. Since the transaction load and the number of secondary copies

grows with the number of sites, the local databases become saturated with propagated updates as the system size increases (except for a very low degree of replication, cf. Fig. 8). This drastically deteriorates throughput and scalability. Consequently, the throughput increases as replication is reduced towards 0%, which means that *no* replication is better than *bad* replication. However, replication might still be required

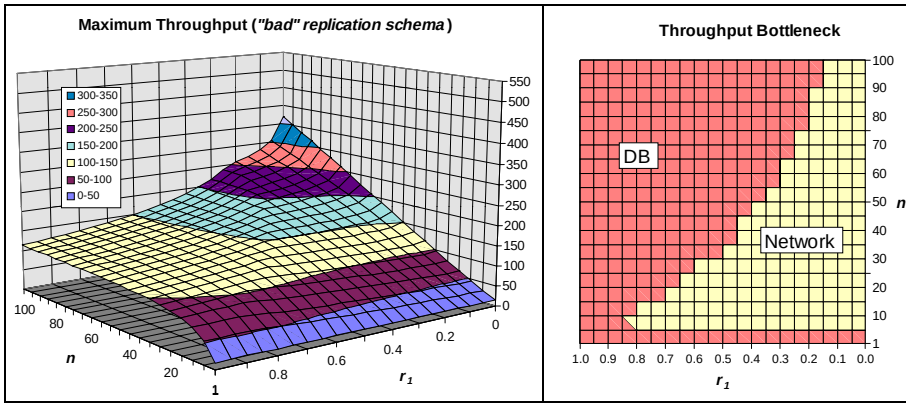

Fig. 8. Throughput $T(r_1,n)$ and Bottleneck Analysis for a bad replication schema

to meet the availability requirements, and results like Fig. 8 clarify the trade-off involved. Some might consider 100 an unreasonably high number of database sites, but the *general* findings represented by Fig. 7 and Fig. 8 do not drastically change if the maximum number of nodes is reduced to 40 or 20 sites, except for absolute values. Furthermore, depending upon the application, the number of sites may indeed range from only a few sites to several hundred sites [3].

6 Summary and Outlook

Based on an analysis of existing performance evaluations, we presented 2RC as an improved modeling approach for performance evaluations of distributed and replicated database systems. 2RC is of increased expressiveness through a new 2-dimensional replication model, an advanced database communication model, the possibility to model the quality of a replication schema, and the ability to consider arbitrary transaction and communication patterns of real-world applications. Particularly, 2RC reflects the close interplay between replica management and intersite communication, and allows for bottleneck considerations. The findings show how partial 2D-replication schemata, which have not been evaluated previously, affect response time, throughput and scalability. Although the results presented here illustrate only a limited number of the possible parameter variations, they demonstrate that 2RC allows more expressive performance evaluations than 1-dimensional approaches with simplifying communication models.

Numerous parameter variations can be investigated in an analytical model as opposed to simulations or measurements, making full validation virtually impossible [2,16,26,6,20]. However, a partial validation of our model has been conducted by matching the results against extensive measurement data gathered in a telecom database prototype [12]. Additionally, we are currently implementing a distributed database testbed to validate the analytical results more thoroughly.

References

[1] G. Alonso: *Partial Database Replication and Group Communication Primitives*, Proceedings of the 2nd European Research Seminar on Advances in Distributed Systems (ERSADS'97), March 1997.

[2] R. Alonso, D. Barbara, H. Garcia-Molina: *Data Caching Issues in an Information Retrieval System*, ACM Transactions on Database Systems, Vol. 15, No. 3, pp. 359-384, 1990.

[3] T. Anderson, Y. Breitbart, H. Korth, A. Wool: *Replication, Consistency, and Practicality: Are These Mutually Exclusive* ?, ACM SIGMOD International Conference on Management of Data, pp. 484-495, June 1998.

[4] F. Bacelli, E.G. Coffmann Jr.: *A database replication analysis using an M/M/m queue with service interruptions*, Performance Evaluation Review, Vol. 11, No. 4, pp. 102-107, 1983.

[5] Sujata Banerjee, Victor O K Li, Chihping Wang: *Performance analysis of the send-on-demand: A distributed database concurrency control protocol for high-speed networks*, Computer Communications, Vol. 17, No. 3, pp. 189-204, March 1994.

[6] A.B. Bondi, V. Jin: *A performance model of a design for a minimally replicated distributed database for database driven telecommunication services*, Journal on Distributed and Parallel Databases, Vol. 4, No. 4, pp. 295-317, October 1996.

[7] Michael J. Carey, Miron Livny: *Distributed Concurrency Control Performance: A Study of Algorithms, Distribution, and Replication*, Proceedings of the 14th International Conference on Very Large Databases, pp. 13-25, 1988.

[8] S. Ceri, M.A.H. Houtsma, A.M.Keller, P.Samarati: *A Classification of Update Methods for Replicated Databases*, Technical Report STAN-CS-91-1392, Stanford University, October 1991.

[9] Shu-Wie Chen, Calton Pu: *A Structural Classification of Integrated Replica Control Mechanisms*, Technical Report CUCS-006-92, Columbia University, NY, 1992.

[10] B. Ciciani, D.M. Dias, P.S. Yu: *Analysis of Replication in Distributed Database Systems*, IEEE Transactions on Knowledge and Data Engineering, Vol. 2, No. 2, pp. 247-261, June 1990.

[11] E.G. Coffmann, Erol Gelenbe, Brigitte Plateau: *Optimization of the number of copies in a distributed system*, IEEE Transactions on Software Engineering, Vol. 7, pp. 78-84, January 1981.

[12] R. Gallersdorfer, K. Klabunde, A. Stolz, M. Essmajor: I*ntelligent Networks as a Data Intensive Application - Final Project Report*, Technical Report AIB-96-14, University of Aachen, 1996.

[13] Rainer Gallersdorfer, Matthias Nicola: *Improving Performance in Replicated Databases through Relaxed Coherency*, Proceedings of the 21th International Conference on Very Large Databases, pp. 445-456, Sept. 1995.

[14] H. Garcia-Molina: *Performance of the Update Algorithms for Replicated Data in a Distributed Database*, Ph.D. Dissertation, revised version, Computer Science Department, Stanford University, North Holland, 1982.

[15] Jim Gray, P. Helland, P. O`Neil, D. Shasha: *The dangers of replication and a solution*, SIGMOD Record, Vol. 25, No. 2, pp. 173-182, June 1996.

[16] S.Y. Hwang, K.S. Lee, Y.H. Chin: *Data Replication in a Distributed System: A Performance Study*, 7th International Conference on Database and Expert Systems Applications, pp. 708-717, 1996.

[17] B. Kemme, G. Alonso: *A Suite of Database Replication Protocols based on Group Communication Primitives*, Proceedings of the 18th International Conference on Distributed Computing Systems, May 1998.

[18] C.S. Keum, E.K. Hong, W.Y. Kim, K.Y. Whang: *Performance Evaluation of Replica Control Algorithms in a Locally Distributed Database System*, Proceedings of the 4th International Conference on Database Systems for Advanced Database Applications, pp. 388-396, April 1995.

[19] L. Kleinrock: *Queueing Systems, Volume I: Theory*, John Wiley & Sons, 1975.

[20] Kin K. Leung: *An Update Algorithm for Replicated Signalling Databases in Wireless and Advanced Intelligent Networks*, IEEE Transactions on Computers, Vol. 46, No. 3, pp. 362-367, March 1997.

[21] D. Liang, S. K. Tripathi: *Performance Analysis of Long-Lived Transaction Processing Systems with Rollbacks and Aborts*, IEEE Transactions on Knowledge and Data Engineering, Vol. 8, No. 5, pp. 802-815, 1996.

[22] M.C. Little, D.L. McCue: *The Replica Management System: A Scheme for Flexible and Dynamic Replication*, Proceedings of the 2nd Workshop on Configurable Distributed Systems, 1994.

[23] J. Mc Dermott, R. Mukkamala: *Performance Analysis of Transaction Management Algorithms for the SINTRA Replicated Architecture Database Systems*, IFIP Transactions (Comp. Science & Technology), Vol. A-47, pp. 215-234, 1994.

[24] Randolph D. Nelson, Balakrishna R. Iyer: *Analysis of a Replicated Database*", Performance Evaluation, Vol. 5, pp. 133-148, 1985.

[25] Esther Pacitti, Erics Simon: *Update Propagation Strategies to Improve Freshness of* Data in Lazy Master Schemes, Technical Report No. 3233, INRIA Rocquencourt, France, August 1997.

[26] J.F. Ren, Y. Takahashi, T. Hasegawa: *Analysis of impact of network delay on multiversion timestamp algorithms in DDBS*, Performance Evaluation, pp. 21-50, July 1996.

[27] D. Saha, S. Rangarajan, S. K. Tripathi: *An Analysis of the Average Message Overhead in Replica Control Protocols*, IEEE Transactions on Parallel and Distributed Systems, Vol. 7, No. 10, pp. 1026-1034, Oct. 1996.

[28] S.H. Son, N. Haghighi: *Performance Evaluation of Multiversion Database Systems*, Proceedings of the 6th International Conference on Data Engineering, pp. 129-136, 1990.

[29] S.H. Son, F. Zhang: *Real-Time Replication Control for Distributed Database Systems: Algorithms and Their Performance*" 4th International Conference on Database Systems for Advanced Database Applications, pp. 214-221, 1995.

[30] M. Stonebraker. *Concurrency Control and Consistency of Multiple Copies of Data in Distributed Ingres*, IEEE Transactions on Software Engineering, 5(3):188-194, 1979.

[31] P. Triantafillou: *Independent Recovery in Large-Scale Distributed Systems*, IEEE Transactions on Software Engineering, Vol. 22, No. 11, pp. 812-826, November 1996.

[32] O. Wolfson, S. Jajodia, Y. Huang: *An Adaptive Data Replication Algorithm*, ACM Transactions on Database Systems, Vol. 22, No. 2, pp. 255-314, 1997.
[33] Shaoyu Zhou, M.H. Williams, H. Taylor: *Practical Throughput Estimation for Parallel Databases*, Software Engineering Journal, Vol. 11, No. 4, pp. 255-263, July 1996.