Integrating Replication and Communication in Performance Models of Distributed Databases [1]

Matthias Nicola Matthias Jarke

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

Abstract

Most performance studies of distributed databases model data replication and intersite communication only in a rudimentary manner. Based on an analysis of existing models, we develop a new *2-dimensional model of replication* and link it to an *intersite communication model* that considers limited network capacity and load-dependent transmission delay. The resulting comprehensive queueing model is a balanced evaluation of both, the database *and* the communication part of a distributed information system, whereas previous models typically emphasized one and neglected the other. This combination allows the analysis of mixed replication schemata which could not be considered in previous studies. It also allows the characterization of situations where the network or the database will be the performance bottleneck in a distributed information system, thus focusing tuning efforts and investment decisions.

1 Introduction

Replication management in distributed databases concerns the decision when and where to allocate physical copies of logical data fragments (placement), when and how to update them to maintain an acceptable degree of mutual consistency (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 can have significant impact on the overall system performance.

The literature offers several algorithms for replica placement [Wolfson et al. 97], [Little, McCue 94], [Acharya, Zdonik 93] as well as replica control [Davidson, Garcia-Molina 85], [Abbott, Garcia-Molina 87], [Ceri et al. 91], [Chen, Pu 92], [Beuter, Dadam 96]. However, the sophistication of such methods is not matched by the existing performance evaluation models. Considering the vast number of alternatives in the design of a replication schema, it becomes apparent that existing performance 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 by most studies. Our aim is to identify and remedy various such flaws in the performance evaluations, in order to derive a model that describes real-world replicated databases more accurately so that new and more expressive results are obtained.

Section 2 briefly surveys performance modeling and evaluation of distributed databases, focusing on the modeling assumptions. Based upon this, section 3 presents a formal classification of how

[1] This work was supported in part by ESPRIT Long Term Research Project DWQ (Foundations of Data Warehouse Quality) and by the Ph.D. Programme "Informatics and Technology" funded by the German DFG at RWTH Aachen. This paper was submitted to the International Conference on Extending Database Technology, EDBT '98.

replication can be modeled. This analysis yields a new *2-dimensional model of replication*. The main technical contribution is in section 4, which develops an analytical queueing model that includes the 2-dimensional replication model as well as a detailed intersite communication model. Section 5 presents some of the most important results derived from our model, including a new bottleneck analysis which is the critical part of any analytical throughput estimation [Zhou et al. 96].

2 Related work

Performance studies of distributed databases employed analytical methods[2] as well as simulations[3]. The major advantage of simulations is that they 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 sensitivity analysis as thoroughly as desired.

Simulation and analytical studies often use queueing systems as the underlying models. Some of the earliest queueing models of distributed databases can be found in [Coffmann et al. 81], [Bacelli, Coffmann 83], [Nelson, Iyer 85], where a fully replicated database of m local sites is modeled by a M/M/m/FCFS queueing system with blocking[4]. Read transactions are processed by the m servers in parallel, while write transactions block all m servers during their service time. 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 as *queueing networks*. [Ciciani et al. 90, 92], [McDermott, Mukkamala 94] use networks of M/M/1 queues so that each local database is modeled as an M/M/1 system. The restriction that all transactions have the same exponentially distributed service times can be relaxed. [Banerjee et al. 94], [Hwang et al. 96] model the local databases as M/G/1 queues with generally distributed service times, while [Gallersdörfer, Nicola 95], [Leung 97] 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 (queries) and updates. Our model in section 4 represents each local database by an $M/H_\tau/1$ queueing system for an arbitrary number of τ different transaction types.

For more details on the database sites, each site can itself be modeled as a queueing network. [Garcia-Molina 82], [Singhal 86], [Cai 87], [Jenq et al. 88] analyse local databases as feedback networks of an I/O-queue and a CPU-queue. Other models include additional local queues, e.g. for lock requests or blocked transactions [Carey, Livny 88], [Cheung et al. 92], [Burger et al. 97]. Such models are of considerable complexity and it usually takes simulations to evaluate them. Lock conflicts and aborts of transactions are usually only found in performance evaluations of concurrency control algorithms[5]. Such models typically have to be evaluated through simulations and are simplified in modeling assumptions concerning replication and communication. Since our model is not primarily intended to compare concurrency control algorithms, we refrain from modeling lock conflicts (yet) in order to focus on replication and communication.

All queueing models of distributed databases assume that transactions arrive as Poisson streams. Most studies also assume uniformly distributed data access, i.e. each data item is accessed with equal probability[6]. Non-uniform data access is more realistic and modeled in [Gallersdörfer, Nicola 95], [Triantafillou 96], [Hwang et al. 96], [Ren et al. 96]. Our model also assumes non-uniform data access and Poisson arrivals.

---

[2] E.g.: [Leung 97], [Ren et al. 96], [Saha et al. 96], [Hwang et al. 96], [Gallersdörfer, Nicola 95], [Alonso et al. 90], [Mukkamala 89]

[3] E.g.: [Burger et al. 97], [Triantafillou 96], [Keum et al. 95], [Son, Zhang 95], [Ulusoy 94], [Cheung et al. 92], [Triantafillou 91], [Ciciani et al. 90, 92], [Son, Kouloumbis 91], [Jenq et al. 89], [Carey, Livny 88], [Noe Andressian 87], [Cai 87], [Barabara, Garcia-Molina 82]

[4] For details on Kendall's classifying notation of queueing systems see [Kleinrock 75], [Jain 91].

[5] E.g.: [Garcia-Molina 82], [Singhal 86], [Jenq et al. 89], [Ciciani et al. 92], [Keum et al. 95], [Ling, Tripathi 96].

[6] E.g.: [Garcia-Molina 82], [Singhal 86, 90], [Jenq et al. 88,89], [Son, Haghighi 90], [Ciciani et al. 92], [Ulusoy, Belford 92], [Cheung et al.92], [Banerjee et al. 94], [McDermott, Mukkamala 94], [Keum et al. 95], [Son, Zhang 95], [Liang, Tripathi 96], [Leung 97]

Most database performance studies model the communication network only implicitly, i.e. by an infinite server that introduces a constant delay for each message, regardless of message size or network load[7]. "Infinite" means unlimited transmission capacity, no queueing of messages, and the network is never considered to be the bottleneck. However, in modern, large wide area applications (Internet) or in wireless and mobile information systems that suffer from low bandwidth, the communication links may indeed become a bottleneck, especially when a large amount of replicas is to be maintained. Very few attempts have been made to combine a detailed database model with a detailed communication model. [Ren et al. 96] represent the communication network by an M/G/∞ system to model generally distributed network delay. In parts of their study, [Alonso et al. 90] assume limited bandwidth and exponentially distributed communication delay by modeling the network as a set of M/M/1 queues. We develop a more detailed communication model in section 4.

3 Classification of Replication in Performance Models

Some performance studies of distributed databases simply assume *no replication*, i.e. each logical data item is represented by exactly one physical copy [Jenq et al. 88, 89], [Son, Haghighi 90], [Ling, Tripathi 96]. Models which consider partial replication rather than full replication either take a logical perspective (*how many objects are replicated?*) or a physical view (*to how many sites are objects replicated?*), but not both. This distinction leads to the following classification, and to the important understanding that *replication models need to consider both dimensions* together:

(1) All objects to all sites (full replication)
Most performance evaluations assume full replication[8], 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 many applications neither full nor no replication is the optimal configuration [Ciciani et al. 90], [Gallersdörfer, Nicola 95].

(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[9]. Formally, the degree of replication can be denoted by a parameter $r \in \{1,2,\ldots,n\}$, 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. Most authors assume that the replicas are distributed evenly across the sites, but it is still undefined *which* copies are placed on *which* sites. We exploit this *undefined replica placement* to model the *quality* of a replication schema (see section 4.3).

(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 used in [Alonso et al. 90] and [Gallersdörfer, Nicola 95]. The some-objects-to-all-sites scheme is orthogonal to the all-objects-to-some-sites approach because the degree of replication is defined along the percentage of data items as opposed to the number of copies. For $r < 1$, the selection of data items to replicate is undefined. This *undefined replica selection* can again be used to model the quality of replication.

---

[7] E.g.: [Garcia-Molina 82], [Singhal 86], [Mariasoosai, Singhal 90], [Triantafillou 91], [Son, Kouloumbis 91], [Ulusoy, Belford 92], [McDermott, Mukkamala 94], [Gallersdörfer, Nicola 95], [Liang, Tripathi 96], [Burger et al. 97], [Hwang et al. 96], [Leung 97].
[8] E.g.: [Coffmann et al. 81], [Garcia-Molina 82], [Barbara, Garcia-Molina 82], [Bacelli, Coffmann 83], [Nelson, Iyer 85], [Singhal 86], [Mariasoosai, Singhal 90], [Son, Kouloumbis 91], [Keum et al. 95], [Son, Zhang 95], [Saha et al. 96].
[9] E.g.: [Noe, Andressian 87], [Carey, Livny 88], [Mukkamala 89], [Ciciani et al. 90,92], [Ulusoy 94], [Burger et al. 97], [Hwang et al. 96], [Triantafillou 96], [Leung 97].

(4) Some objects to some sites (2-dimensional partial replication)

The two orthogonal 1-dimensional replication models can by combined to a new 2-dimensional scheme: 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, i.e. are represented by only one physical copy. No replication is expressed by $r_1 = 0$. Full replication is modeled by ($r_1 = 1$, $r_2 = n$). This 2D-model owns the property of *undefined replica selection and placement*, since neither the selection of data items to replicate nor their placement is defined. Both aspects can be used to model the quality of 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 calculations) independently from d and normalized to the interval [0;1] as an overall level of replication. This is achieved through dividing by $d \cdot n$, yielding $(r_1 \cdot r_2)/n$.

This characteristic of the 2D-approach is depicted in figure 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). The orthogonal some-objects-to-all-sites scheme studies replication along the line from (1,50) to (0,50). Using the 2D-scheme, any point in Figure 1 is a possible replication schema.

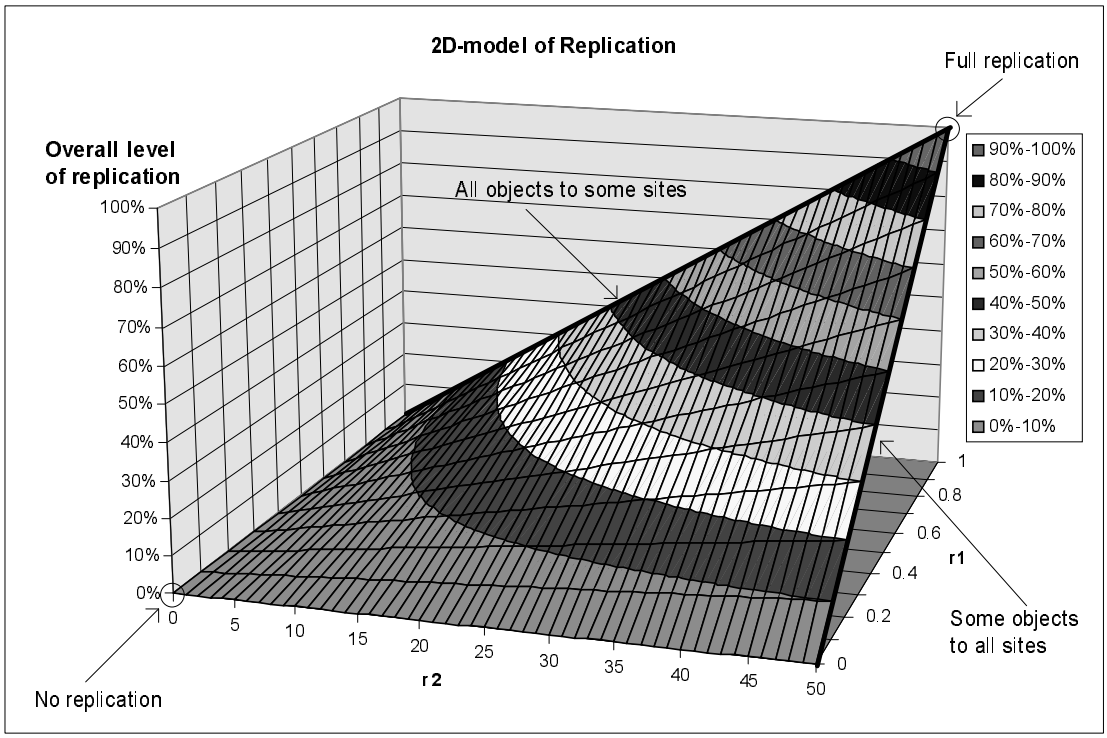

Figure 1: The 2-dimensional model of replication

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. Furthermore, 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 cannot be modeled with the all-objects-to-some-sites scheme. The 2D-approach can capture such scenarios and models realistic replication schemata more accurately than previous studies. Thus, we believe that integrating the 2D-scheme in an analytical queueing model is a profitable contribution towards a better understanding of how replication affects distributed system performance. We give some example of such enhanced strategies and their evaluation in section 5.

4 A Queueing Model of 2D-Replication and Intersite Communication

This section presents our queueing model of a replicated database. In secton 4.1, a graphical visualization provides an overview and a comparison to other studies. Sections 4.2 through 4.5 present the mathematical details. Some of the more complex formulas are derived in an Appendix..

4.1 Overview and comparison with other models

Figure 2 sketches the dependencies we consider necessary to capture in an accurate performance model. 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.

τ transaction types and 2 message types per transaction (with individual transmission times depending on the message size) allow to model a wide range of different applications and workload patterns. 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. The two bold arrows highlight the important dependencies through which load dependent communication delay and network dependent throughput are captured. This combines a comprehensive database and replication model with a detailed communication model, and covers the interdependencies between the two more accurately than previous studies (Figure 3)[10].

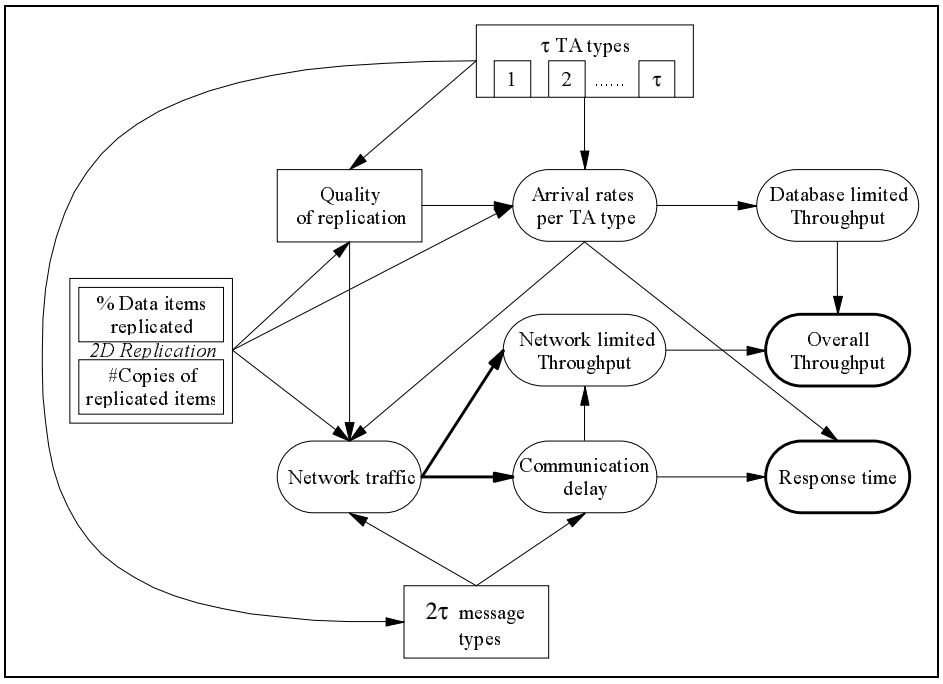

Figure 2: Graphical representation of our model.

Although more advanced than many others, the model in [Hwang et al 96] assumes constant communication delay and unlimited network capacity. On the database side, they consider 1D-replication (all-objects-to-some-sites) and do not allow different types of queries or updates (like all the models in figure 3).

[Alonso et al. 90] have a more sophisticated communication part which captures load dependent communication delay. However, this is not exploited for throughput calculation and combined with a simple database model of 1D-replication (some-objects-to-all-sites) and only 2 transaction types. [Hwang et al 96], [Alonso et al. 90] model non-uniform data access by a factor for access locality and hot spot access, respectively.

[10] Unfortunately, many studies do not provide sufficient details of their models to allow a characterisation like in figure 3. Moreover, note that the figures 2 and 3 are quite simplified representations of the corresponding models.

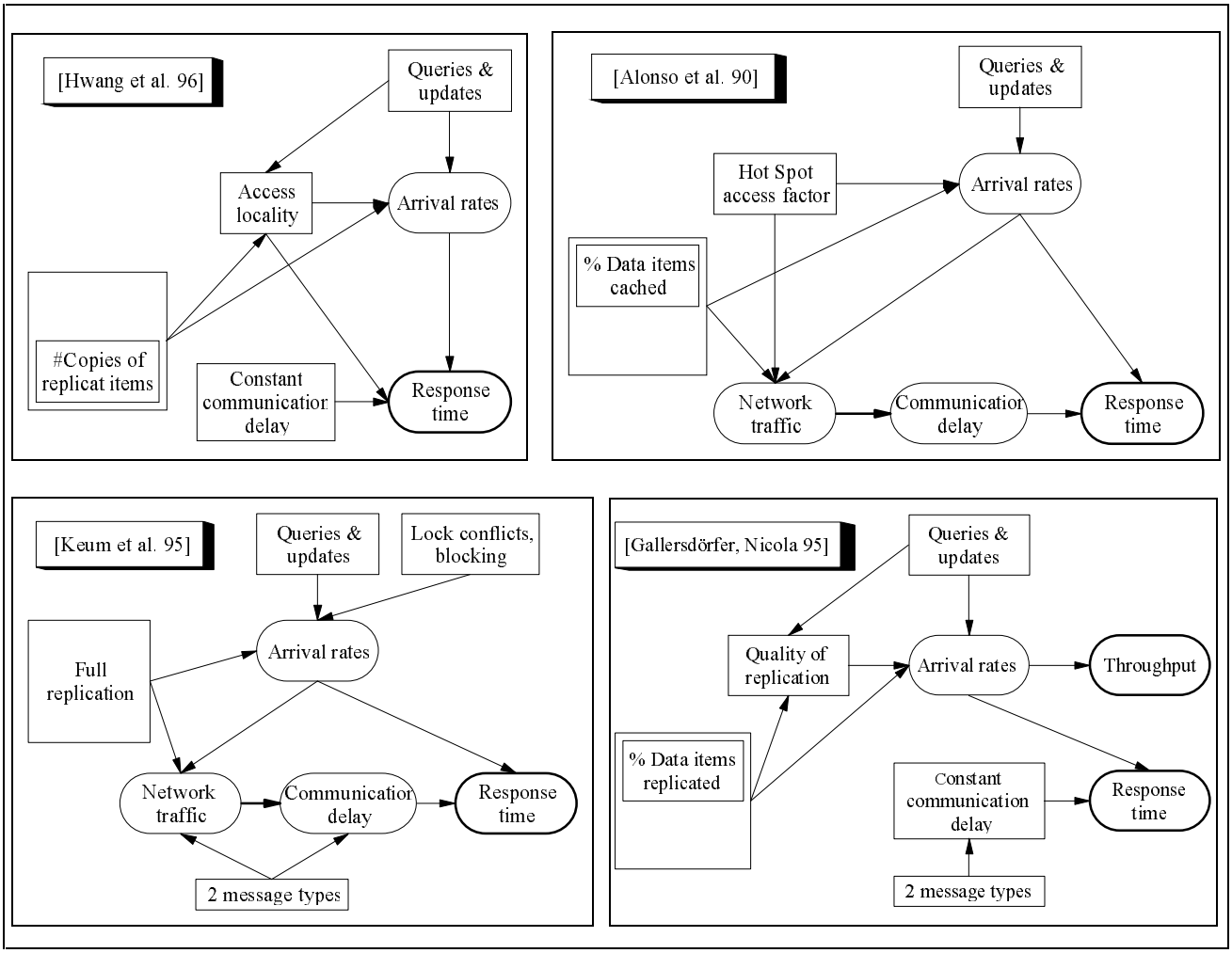

Figure 3: Graphical representations of other models.

The model in [Keum et al. 95] considers 2 different message types, lock conflicts and blocking, but suffers from a very simple replication model and the assumption of uniform data access.
[Gallersdörfer, Nicola 95] is one of the few analytical studies that model the quality of replication and calculate throughput as a performance criterion. However, they assume constant communication delay and unlimited network capacity which leads to a more simplistic throughput estimation than in our model. Moreover, they only consider 1D-replication (some-objects-to-all-sites) and a simple workload model, which implies a rudimentary model of the quality of replication. Unlike previous studies, our model integrates 2D-replication, non-uniform data access and arbitrary workloads with network dependent throughput calculation, which allows detailed bottleneck considerations. We conclude that our model is not just an increment on any existing study, but a unifying framework for thorough distributed database performance evaluation.

4.2 Workload and transaction processing

We model a replicated database by an open queueing network[11] in which each of the n identical sites is represented by a $M/H_\tau/1$ system. Transaction arrivals to the distributed system from outside are modeled by n identical Poisson processes with parameter λ (one arrival stream per site), because Poisson streams have been found to be a good approximation for jobs submitted independently by a large number of users [Kleinrock 75], [Jain 75]. The τ different types of transactions are numbered

[11] Open queueing networks have also been used in [Mc Dermott, Mukkamala 94], [Mariasoosai, Singhal 90], [Singhal 86], [Garcia-Molina 82], [Jenq et al. 89], [Barbara, Garcia-Molina 82] as opposed to closed queueing networks in [Carey, Livny 88], [Liang, Tripathi 96].

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:

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

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

Transactions are assumed to be processed according to the primary copy approach[12], and updates are propagated asynchronously to the secondary copies. 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 [Gallersdörfer, Nicola 95]. Small values of k express high relaxation, $k = 0$ models suspended update propagation, and for $k = 1$ updates are propagated immediately.

4.3 Locality and quality of replication

Without replication but due to skilful data allocation, transactions exhibit a behaviour of locality in the sense that 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 [Wolfson et al. 97]. Thus, it is desirable to 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: Since updates have to be executed on the primary copy, only the place of the primary copy rather than the placement of the secondary copies matters. However, the replica selection is of much higher significance: selecting many update intensive data items for replication causes high update propagation overhead. Thus, 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 data items selected for replication 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: For query processing, replica placement is much more crucial than replica selection: Even if all read itensive 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

[12] See [Stonebreaker 79] for details. The primary copy scheme has been judged advantageous over other replica control concepts in various studies ([Gray et al. 96], [Keum et al. 95]) and is implemented in commercial systems like Sybase and Oracle.

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. Thus, sel_i and $plcmt_i$ have to depend on the degree of replication and are defined as

$$plcmt_i = \frac{1}{\left(\frac{r_1 \cdot r_2}{n}\right)^{f_plcmt_i}} \qquad and \qquad sel_i = \frac{1}{(r_1)^{f_sel_i}}$$

so that the parameters $f_plcmt_i \in [0,1]$ and $f_sel_i \in [-\infty,1]$ can „*fine-tune*“ how far or close sel_i and $plcmt_i$ follow their optimum values $1/r_1$ and $n/(r_1 \cdot r_2)$ respectively (see Appendix A)

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

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

as argued in Appendix B. Note, that loc_i, sel_i and $plcmt_i$ model non-uniform data access. The total arrival rate of transactions at a site is the sum of the total arrival rates of all transaction types i :

$$\lambda^{total} = \sum_{i=1}^{\tau} \lambda_i^{total} \qquad with \qquad \lambda_i^{total} = \lambda_i + (1 - q_i) \cdot (r_1 \cdot sel_i) \cdot (r_2 - 1) \cdot k \cdot \lambda_i$$

which are transactions from outside, from other sites, and propagated updates (see Appendix C).

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. We assume that for each transaction type i the communication delay for a *send* (*return*) is 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 $t_c^{return_i}$ of the exponentially distributed service times are hence defined as

$$t_c^{send_i} = \frac{8 \cdot size_c^{send_i}}{baud} \qquad and \qquad t_c^{return_i} = \frac{8 \cdot size_c^{return_i}}{baud}$$

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

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

The first term covers transactions forwarded to remote sites due to a lack of appropriate local data as well as returned query results, the second term covers update propagation (see Appendix D). 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 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 amount 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 $t_c^{return_i}$ ($1 \le i \le \tau$.). As explained in Appendix D, the mean waiting time $\overline{W}_c$ at a communication server results in

$$\overline{W}_c = \frac{\sum_{i=1}^{\tau} \left((1-\ell_i)\cdot\lambda_i + (1-q_i)\cdot(r_1\cdot sel_i)\cdot(r_2-1)\cdot k\cdot\lambda_i\right)\cdot\left(t_c^{send_i}\right)^2 + q_i\cdot(1-\ell_i)\cdot\lambda_i\left(t_c^{return_i}\right)^2}{1-\sum_{i=1}^{\tau}\left((1-\ell_i)\cdot\lambda_i + (1-q_i)\cdot(r_1\cdot sel_i)\cdot(r_2-1)\cdot k\cdot\lambda_i\right)\cdot t_c^{send_i} + q_i\cdot(1-\ell_i)\cdot\lambda_i\cdot t_c^{return_i}}$$

4.5 Performance criteria

We consider the average response times and the transaction throughput as performance criteria. The mean waiting time 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*. Details on the derivations are provided in Appendix E.

In steady state, the throughput of the local databases equals the arrival rate λ but is bounded 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 (see Appendix F) 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 Applications of the Model

In this section, we demonstrate that our 2D approach can recommend advantageous new replication strategies that cannot be evaluated by previous models. We present a selection of results, based on the parameter values shown in table 1 unless otherwise stated. Parameters not listed in table 1 can be calculated from the values given. The base settings are carefully chosen after measurements in database systems for telecom applications [Gallersdörfer et al. 97]. The values also agree with those found in [Alonso et al. 90], [Ciciani et al. 90]. Although we will mention absolute values to refer to characteristics in the diagrams below, we consider the general trends and shapes of the graphs as the primary results. We consider 3 transaction types: short queries, updates, and long queries (e.g. statistical evaluations). The sensitivity analysis cannot be completely discussed here. It shows that parameter variations affect the performance values in a reasonable way, i.e. the model is stable.

Parameter	***Meaning***	***Base setting***
n	Number of sites	50
λ	Transaction arrival rate per site (TPS)	3 (i.e. 150 TPS total)
τ	Number of transactions types	3
q_i	Function to distinguish between queries and updates	$q_1 = 1$, $q_2 = 0$, $q_3 = 1$
a_i	Percentage of transactions of type i	$a_1 = 0.85$, $a_2 = 0.1$, $a_3 = 0.05$
r_1	Percentage of data items replicated	varying
r_2	Number of copies of replicated data items	varying
loc_i	Transactions' locality without replication	0.04
f_plcmt_i	Fine tuning replica placement	0.55
f_sel_i	Fine tuning replica selection	-0.8
k	Coherency index	1
t_i	Mean service time for a transaction of type i (seconds)	$t_1 = 0.06$, $t_2 = 0.125$, $t_3 = 0.5$
$baud$	Communication bandwidth (bits per second)	64000 (e.g. ISDN)
$size_c^{send_i}$	Message size for a *send* of a transaction of type i.	400 byte each
$size_c^{return_i}$	Message size of returned query results (byte)	$size_c^{return_1} = 2000$, $size_c^{return_3} = 3500$

Table 1: Parameter base values

Figure 4 shows the combined average response time $\overline{\mathcal{R}}$ over the r_1-r_2-*space*. A moderate degree of replication ($r_1 < 0.5$, $r_2 < 20$) leads to increased local 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 individually satisfy an application's reliability and availability requirements while retaining similarly low response times. This result shows the increased expressiveness of the 2D-model over 1-dimensional approaches.

Figure 5 illustrates the maximum throughput $T(r_1,r_2) = min(T_D, T_C)$. 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. 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 more data items

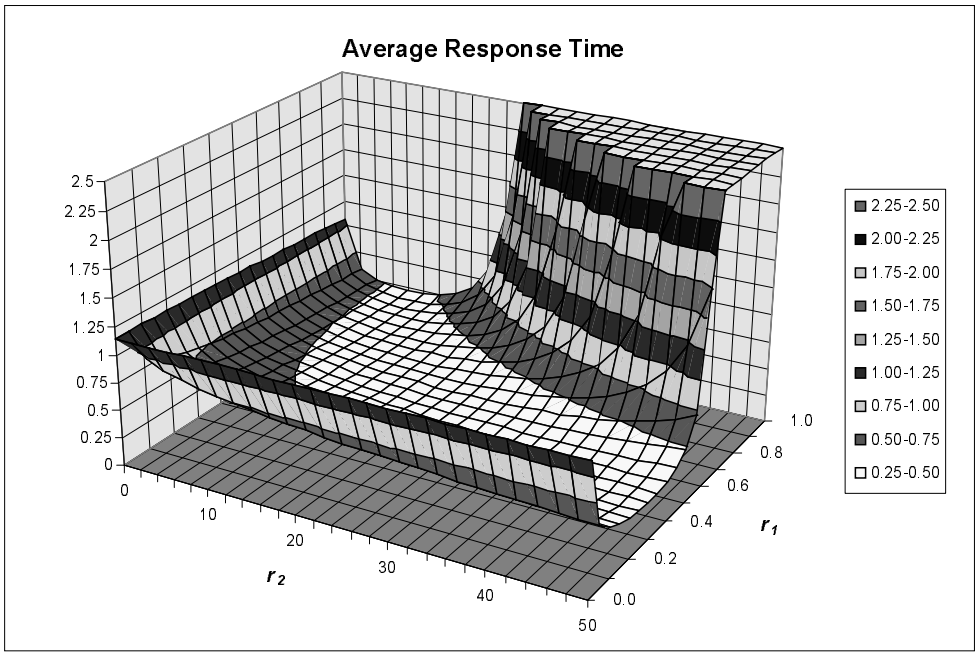

Figure 4: Average overall response time $\overline{R}(r_1, r_2)$

should be replicated (e.g. 50%), a medium number of copies yields the maximum throughput (e.g. $r_2 = 30$ for $r_1 = 0.5$). For applications that need to trade throughput for a high availability, it is worthwhile to know that replicating 75% of the data items to 40 sites allows a twice as high throughput as full replication.

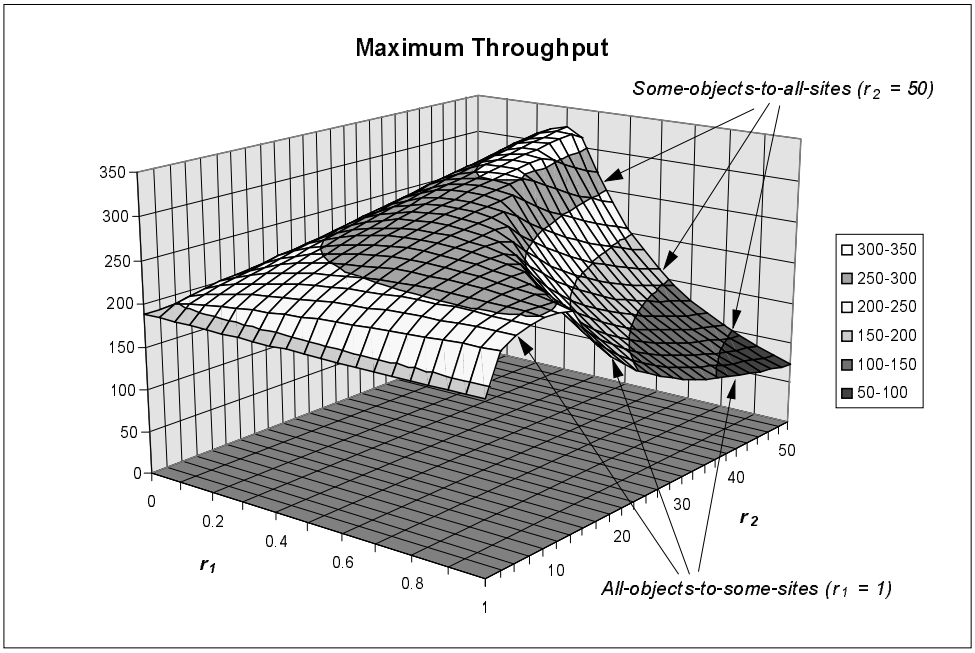

Figure 5: Overall system throughput

Since our throughput calculation considered limited network capacity, we can determine for any replication schema whether the communication network or the local databases are the throughput bottleneck (Figure 6). 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.

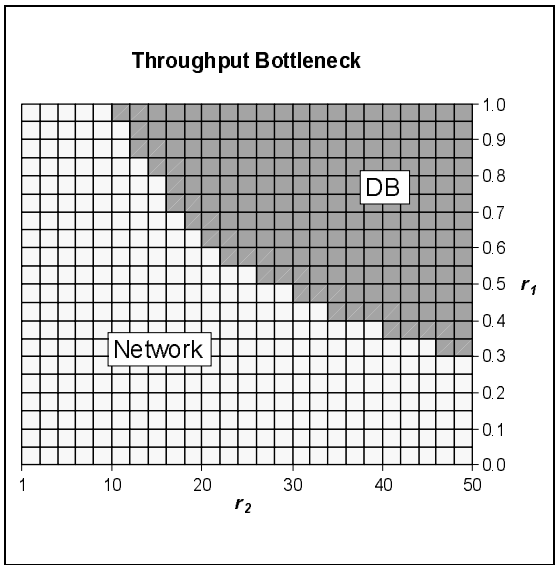

Figure 6: Bottleneck Analysis

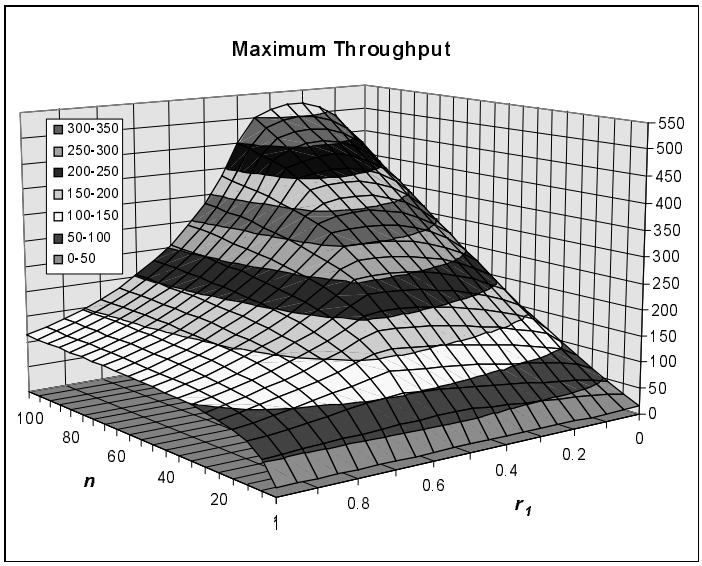

Figure 7 Overall throughput $T(r_1,n)$

Figure 7 depicts the throughput as a function of r_1 and the system size n. For any value of n, r_2 is set to $2n/3$, i.e. r_1% of the data items are replicated to 2/3 of the 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. 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 (Figure 8) and throughput can grow to over 500 TPS for $r_1 = 0.25$.

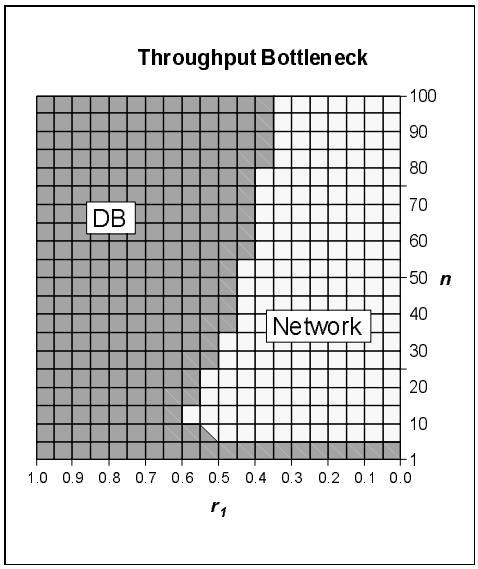

Figure 8

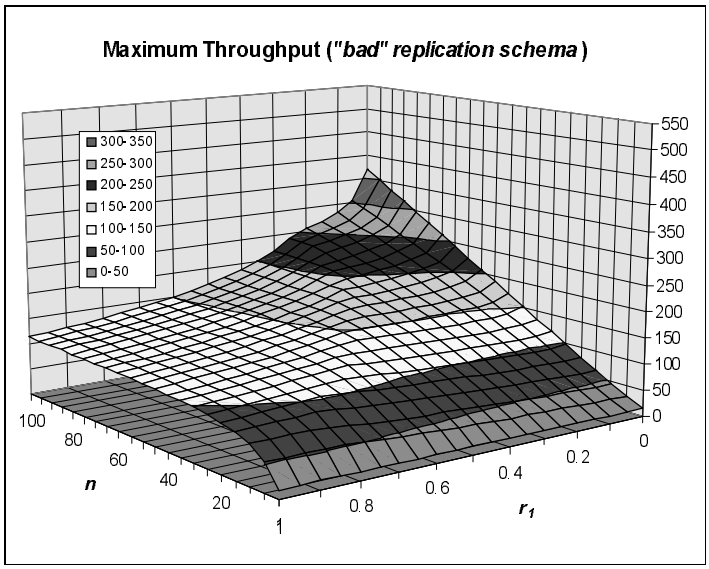

Figure 9

In figure 9 we examine the impact of the quality of replication on the throughput by considering a replication schema in which update intensive data is replicated to sites where it is not read very often ($f_plcmt_i = 0$, $f_sel_i = 0.5$). Such a replication schema causes substantial overhead which 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.

6 Summary and Outlook

Based on the analysis of existing performance evaluations, we presented a new 2-dimensional replication model which covers a much larger space of possible replication schemata. The 2D-replication model has been integrated into an analytical queueing model of replicated databases, which covers an arbitrary number of transaction and message types, non-uniform data access, relaxed coherency, and load dependent communication delay. With a graphical representation, we clarified that our model captures more complex dependencies than other studies.

The results show how partial 2D replication schemata, which have not been considered previously, affect response time and throughput. This allows performance estimations for distributed databases where an arbitrary number of data items is replicated to an arbitrary number of sites, which is a more common and realistic case than replicating *some-objects-to-all-sites* or *all-objects-to-some sites*, as assumed in other performance evaluations. Our model also reflects the close interplay between replica management and intersite communication. Thus, our bottleneck analysis revealed, for which replication schemata or system size the network or the local databases are the throughput limiting factor. This work is currently extended to consider various concurrency control aspects like lock conflicts and blocking of transactions.

Numerous parameter variations can be investigated in an analytical model as opposed to simulations or measurements. Consequently, a validation through simulations or measurements in all dimensions of the parameter space is hardly feasible, like in other analytical studies, e.g. [Alonso et al. 90], [Hwang et al. 96], [Ren et al. 96], [Bondi, Jin 96], [Leung 97]. Nevertheless, we are currently implementing a distributed database testbed to validate some of our analytical results through measurements in the context of wireless information systems. One application we are focusing on is mobile data caches for data warehouses, another one is distributed health care management in developing countries [Nicola, Jarke 97].

References

[Abbott, Garcia-Molina 87] R. Abbott, H. Garcia-Molina: „*Reliable Distributed Database Management*“, Proceedings of the IEEE, 75 (5), pp. 601-620, May 1987.

[Acharya, Zdonik 93] Swarup Acharya, Stanley B. Zdonik: „*An Efficient Scheme for Dynamic Data Replication*“, Technical Report CS-93-43, Brown University, September 1993.

[Alonso et al. 90] R. Alonso, D. Barbará, H. Garcia-Molina: „*Data Caching Issues in an Information Retrieval System*“, ACM Transactions on Database Systems, Vol. 15, No. 3, pp. 359-384, 1990.

[Bacelli, Coffman 83] F. Bacelli, E.G. Coffmann Jr.: „*A database replication analysis using an M/M/m queue with service interruptions*“, Performance Evaluation Review, Vol. 11, Nr.4, pp. 102-107, 1983.

[Banerjee et al. 94] 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.

[Barbará, Garcia-Molina 82] D.Barbara, H. Garcia-Molina: „*How Expensive is Data Replication? An Example.*“, 2nd International Conference on Distributed Computing Systems, pp. 263-268, February 1982.

[Beuter, Dadam 96] T. Beuter, P. Dadam: „*Principles of replication control in distributed database systems*“ (in German), Informatik Forschung und Technik, Vol. 11, No. 4, pp. 203-212, 1996.

[Bondi, Jin 96] 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.

[Burger et al. 97] A. Burger, V. Kumar, M.L. Hines: „*Performance of Multiversion and Distributed Two-Phase Locking Concurrency Control Mechanisms in Distributed Database*“, Information Sciences, Vol. 96, No. 1-2, pp. 129-157, January 1997.

[Cai 87] Jian Cai: „*Simulation and Evaluation of Distributed Database Systems*“, Informatik Fachberichte 154, pp. 313-326, October 1987.

[Carey, Livny 88] Michael J. Carey, Miron Livny: „*Distributed Concurrency Control Performance: A Study of Algorithms, Distribution, and Replication*“, 14th Conference on Very Large Databases, pp. 13-25, 1988.

[Ceri et al. 91] 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.

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

[Cheung et al. 92] S.Y. Cheung, M.H. Ammar, M. Ahamad: „*The Grid Protocol: A High Performance Scheme for Maintaining Replicated Data*“, IEEE Transactions on Knowledge and Data Engineering, Vol. 4, No. 6, pp. 582-592, December 1992.

[Ciciani et al. 90] 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.

[Ciciani et al. 92] B. Ciciani, D.M. Dias, P.S. Yu: „*Analysis of Concurrency-Coherency Control Protocols for Distributed Transaction Processing Systems with Regional Locality*“, IEEE Transactions on Software Engineering, Vol. 18, No. 10, pp. 899-914, October 1992.

[Coffmann et al. 81] 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.

[Davidson, Garcia-Molina 85] Susan B. Davidson, Hector Garcia-Molina: „*Consistency in Partitioned Networks*“, Computing Surveys 17(3), pp. 341-370, 1985.

[Gallersdörfer, Nicola 95] Rainer Gallersdörfer, Matthias Nicola: „*Improving Performance in Replicated Databases through Relaxed Coherency*“, 21th Conference on Very Large Databases, pp. 445-456, September 1995.

[Gallersdörfer et al. 97] Rainer Gallersdörfer, Matthias Nicola, Matthias Jarke: „*The ADR Replication Manager: Design, Implementation, and Evaluation in Telecom Applications*“, submitted for publication.

[Garcia-Molina 82] H. Garcia-Molina: „*Performance of the Update Algorithms for Replicated Data in a Distributed Database*“, Ph.D. Dissert., revised, Computer Science Dept., Stanford University, North Holland, 1982.

[Gray et al. 96] 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.

[Hwang et al. 96] 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.

[Jain 91] R. Jain: „*The Art of Computer Systems Performance Analysis - Techniques for Experiment Design, Measurement, Simulation and Modeling*“, John Wiley & Sons, 1991.

[Jenq et al. 88] Bao Chyuan Jenq, W.H. Kohler, D. Towsley: „*A Queueing Network Model for a Distributed Database Testbed System*“, IEEE Transactions on Software Engineering, Vol. 14, No. 7, pp. 908-921, July 1988.

[Jenq et al. 89] B.C. Jenq, B.C. Twitchell, T.W. Keller: „*Locking Performance in a Shared Nothing Parallel Database Machine*“, IEEE Transactions on Knowledge and Data Engineering, Vol. 1, No. 4, pp. 530-543, Dec. 1989.

[Keum et al. 95] 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.

[Kleinrock 75] Leonard Kleinrock: „*Queueing Systems, Volume I: Theory*“, John Wiley & Sons, 1975.

[Leung 97] 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.

[Liang, Tripathi 96] D. Liang, S. K. Tripathi: „*Performance Analysis of Long-Lived Transaction Processing Systems with Rollbacks and Aborts*“, IEEE Trans. on Knowledge and Data Engineering, Vol. 8, No. 5, pp. 802-815, 1996.

[Little, McCue 94] 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, March 1994.

[Mariasoosai, Singhal 90] W. Mariasoosai, M Singhal: „*A Concurrency Control Algorithm for Replicated Database Systems*“, Proceedings of the ISMM International Conference, New York, pp. 143-147, October 1990.

[Mc Dermott, Mukkamala 94] John Mc Dermott, Ravi Mukkamala: „*Performance Analysis of Transaction Management Algorithms for the SINTRA Replicated Architecture Database Systems*“, IFIP Transactions (Computer Science and Technology), Vol. A-47, pp. 215-234, 1994.

[Mukkamala 89] Ravi Mukkamala: „*Measuring the Effects of Data Distribution Models on Performance Evaluation of Distributed Database Systems*“, IEEE Transactions on Knowledge and Data Engineering, Vol.1, No.4, pp. 494-507, December 1989.

[Nelson, Iyer 85] Randolph D. Nelson, Balakrishna R. Iyer: „*Analysis of a Replicated Database*“, Performance Evaluation, Vol. 5, pp. 133-148, 1985.

[Nicola, Jarke 97] Matthias Nicola, Matthias Jarke: „*Design and Evaluation of Wireless Health Care Information Systems in Developing Countries*“, Technical Report AIB-97-6, Technical University of Aachen,

[Noe, Andreassian 87] J.D. Noe, A. Andreassian: „*Effectiveness of Replication in Distributed Computer Networks*“, Proceedings of the 7th International Conference on Distributed Computing Systems, pp. 508-513, 1987.

[Ren et al. 96] 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.

[Saha et al. 96] 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.

[Singhal 86] Mukesh Singhal: „*Concurrency Control Algorithms and their Performance for Replicated Database Systems*“, Ph.D. dissertation, Department of Computer Science, University of Maryland, 1986.

[Singhal 90] Mukesh Singhal: „*Update Transport: A New Technique for Update Synchronization in Replicated Database Systems*“, IEEE Transactions on Software Engineering, Vol. 16, pp. 1325-1336. 1990.

[Son, Haghighi 90] S.H. Son, N. Haghighi: „*Performance Evaluation of Multiversion Database Systems*“, Proceedings of the 6th International Conference on Data Engineering, pp. 129-136, February 1990.

[Son, Kouloumbis 91] S.H. Son, S. Kouloumbis: „*Performance Evaluation of Replication Control Algorithms for Distributed Database Systems*“, Technical Report, CS-TR-91-11, University of Virginia, 1991.

[Son, Zhang 95] 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, April 1995.

[Triantafillou 91] P. Triantafillou: „*Employing Replication to Achieve High Availability and Efficiency in Distributed Systems*“, Research Report CS-91-28, University of Waterloo, 1991.

[Triantafillou 96] Peter Triantafillou: „*Independent Recovery in Large-Scale Distributed Systems*“, IEEE Transactions on Software Engineering, Vol. 22, No. 11, pp. 812-826, November 1996.

[Ulusoy 94] Ö. Ulusoy: „*Processing Real-Time Transactions in a Replicated Database System*“, Journal on Distributed and Parallel Databases, Vol. 2, No. 4, pp. 405-436, October 1994.

[Ulusoy, Belford 92] Ö. Ulusoy, G.G. Belford: „*A Simulation Model for Distributed Real-Time Databases*“, Proceedings of the 25th Annual Simulation Symposium (IEEE), pp. 232-240, April 1992.

[Wolfson et al. 97] Ouri Wolfson, S. Jajodia, Y. Huang: „*An Adaptive Data Replication Algorithm*“, ACM Transactions on Database Systems, Vol. 22, No. 2, pp. 255-314, June 1997.

[Zhou et al. 96] 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.

Appendix

This appendix contains some additional explanations on the mathematical model and a more detailed derivation of the performance values presented in section 4.

A. Quality of replication

To provide more insight into the model of the quality of replication, table 2 summarizes the related parameters, their values and the associated meanings concerning replica selection and replica placement.

	Range	best selection	unbiased	worst
f_sel_i	$[-\infty, 1]$	$\rightarrow -\infty$	0	1
sel_i	$[0;1/r_1]$	0	1	$1/r_1$
	Range	best placement	unbiased/bad	
f_plcmt_i	[0,1]	1	0	
$plcmt_i$	$[1; n/(r_1 \cdot r_2)]$	$n/(r_1 \cdot r_2)$	1	

Table 2: Parameters modeling the quality of replication

Since replica placement can never decrease the local read probability, the interval of values for *plcmt* starts at 1 instead of 0. Note, that that for $plcmt = 1$ the „worst possible“ replica placement is still a quite good one when the overall degree of replication is high. Mathematically, this is because the optimum value for *plcmt* (i.e. $n/(r_1 \cdot r_2)$) converges towards 1 as the degree of replication increases. Hence $plcmt = 1$ „gets closer“ to the optimum replica placement. Intuitively, the higher the degree of replication the more difficult it is to design a „bad“ replica placement. Similarly, the higher the degree of replication the more difficult it is to design a „bad“ replica selection: if sel_i was set to $1/r_1$ in order to define a bad replica selection, sel_i necessarily becomes 1 as the percentage of replicated data items (i.e. r_1) converges towards 1. Intuitively, if *all* data items are selected for replication, there is no way one could *only* pick the update intensive data items. Hence, if *all* data items are selected for replication, the replica selection is necessarily unbiased.

B. Locality

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, or equivalently: a share of loc_i of the arriving transactions can run at the local site (first term). With probability $1 - loc_i$, queries ($q(i) = 1$) cannot be executed locally without replication, but replication increases the local read availability by $(r_1 \cdot r_2)/n$ or higher, depending on the quality ($plcmt_i$) of the replication schema (second term).

If $plcmt_i = 1$, i.e. „unbiased“ replica placement, the impact of replication on ℓ_i is barely expressed by the extent of replication, namely $(r_1 \cdot r_2)/n$. If $plcmt_i > 1$, then

$$\frac{r_1 \cdot r_2}{n} \cdot plcmt_i > \frac{r_1 \cdot r_2}{n}$$

which means that the positive influence of replication on ℓ_i is greater than the mere extent of replication, because the replica placement is advantageous. For $plcmt_i = n/(r_1 \cdot r_2)$, i.e. optimum replica placement, it follows that

$$\frac{r_1 \cdot r_2}{n} \cdot plcmt_i = 1$$

which means that even if there is no full replication, the replica placement is so fortunate that queries can always be answered locally. Hence, in the model a partial replication schema with optimum replica placement has the same effect on ℓ_i as full replication with unbiased replica placement.

C. The arrival rate

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

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

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. If transactions of type i are updates (i.e. $1 - q(i) = 1$) the local total arrival rate λ_i^{total} is increased by the number of transactions of type i that arrive due to 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

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

The probability $r_1 \cdot sel_i$ that an update hits a replicated data item which is replicated, is r_1 for unbiased replica selection ($sel_i = 1$). It is 0 if an optimal replica selection managed not to replicate data items which are subject to modifications ($sel_i = 1$), and it is 1 if updates always hit replicated data items ($sel_i = 1/r_1$). Hence, in the model a partial replication schema with the worst replica selection has the same effect on λ_i^{total} as full replication with unbiased replica selection.

The parameters ℓ_i and n disappeared from λ_i^{total}, which indicates that every local database receives just as many transactions from the remote sites as it sends out to them, because the sites are assumed to behave identical and intersite communication is symmetrical.

D. Intersite communication

The average number of messages per second $\overline{N}$ in the distributed system amounts to

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

out of which there are

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

messages of type *send* (the first term covers transactions forwarded to remote sites due to a lack of appropriate local data, the second term covers update propagation) and

$$\sum_{i=1}^{\tau} q_i\cdot n\cdot(1-\ell_i)\cdot\lambda_i$$

messages of type *return*, which are returned query results. These results follow straight from the transaction arrival rates. The factor n is required to capture the messages generated by n sites and not only 1 site. It follows that a share of

$$\frac{n\cdot(1-\ell_i)\cdot\lambda_i \quad + \quad (1-q_i)\cdot n\cdot(r_1\cdot sel_i)\cdot(r_2-1)\cdot k\cdot\lambda_i}{\overline{N}}$$

of the $\overline{N}$ messages has mean service time $t_c^{send_i}$ (for $1 \le i \le \tau$) and a share of

$$\frac{q_i\cdot n\cdot(1-\ell_i)\cdot\lambda_i}{\overline{N}}$$

of the messages has mean service time t_c^{return} (for $1 \le i \le \tau$). Hence, the first and second moment of our $H_{2\tau}$ service time distribution can be derived following [Kleinrock 75], p.142, and result in

$$E(B_c) = \sum_{i=1}^{\tau} \frac{n\cdot(1-\ell_i)\cdot\lambda_i \quad + \quad (1-q_i)\cdot n\cdot(r_1\cdot sel_i)\cdot(r_2-1)\cdot k\cdot\lambda_i}{\overline{N}}\cdot t_c^{send_i}$$

$$+ \quad \sum_{i=1}^{\tau} \frac{q_i\cdot n\cdot(1-\ell_i)\cdot\lambda_i}{\overline{N}}\cdot t_c^{return_i}$$

and

$$E(B_c^2) = \sum_{i=1}^{\tau} \frac{2\cdot n\cdot(1-\ell_i)\cdot\lambda_i \quad + \quad 2\cdot(1-q_i)\cdot n\cdot(r_1\cdot sel_i)\cdot(r_2-1)\cdot k\cdot\lambda_i}{\overline{N}}\cdot (t_c^{send_i})^2$$

$$+ \quad \sum_{i=1}^{\tau} \frac{2\cdot q_i\cdot n\cdot(1-\ell_i)\cdot\lambda_i}{\overline{N}}\cdot (t_c^{return_i})^2$$

respectively[13]. Since a $M/H_{2\tau}/1$ system is a special case of a M/G/1 system, 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 ([Kleinrock 75] p.187-191):

$$\overline{W} = \frac{\lambda \cdot E(B^2)}{2(1-\rho)}$$

Considering that the utilization ρ can be expressed as $\rho = \lambda_c \cdot E(B_c)$ where the arrival rate is here $\lambda_c = \overline{N}/n$, the mean waiting time amounts to

$$\overline{W}_c = \frac{\frac{\overline{N}}{n} \cdot \frac{E(B_c^2)}{2}}{1 - \frac{\overline{N}}{n} \cdot E(B_c)}$$

$$= \frac{\sum_{i=1}^{\tau} \left((1-\ell_i)\cdot\lambda_i + (1-q_i)\cdot(r_1 \cdot sel_i)\cdot(r_2-1)\cdot k\cdot\lambda_i\right)\cdot\left(t_c^{send_i}\right)^2 + q_i \cdot (1-\ell_i)\cdot\lambda_i (t_c^{return_i})^2}{1 - \sum_{i=1}^{\tau} \left((1-\ell_i)\cdot\lambda_i + (1-q_i)\cdot(r_1 \cdot sel_i)\cdot(r_2-1)\cdot k\cdot\lambda_i\right)\cdot t_c^{send_i} + q_i \cdot (1-\ell_i)\cdot\lambda_i \cdot t_c^{return_i}}$$

E. Database service time, waiting time, and response time

Usually, the time required to process a transaction locally is mainly determined by the disk service time, which in turn is closely related to the number of data objects referenced [Son, Haghighi 90]. Transactions that access a small or moderate number of data items are expected to occur more frequently than transactions that reference a large number of data objects. This can be expressed if the number of data objects accessed per transaction are assumed to be geometrically distributed. The service time for a transaction of type *i* at a local database is then modeled as exponentially distributed with mean t_i (seconds), which is a „continuous version" of the discrete geometric distribution. Hence, the service time for the combined arrival process of all τ transaction types follows an τ-phase hyperexponential distribution [Kleinrock 95], [Jain 91].

However, the service time for the combined arrival process of all τ transaction types is no longer exponentially distributed but follows an τ-stage hyperexponential distribution [Kleinrock 95], [Jain 91]. This means that the service time for a transaction from the combined flow is with probability $\lambda_i^{total}/\lambda^{total}$ „taken from" an exponential distribution with mean t_i. This leads to the first and second moment of the H_τ service time distribution as

$$E(B) = \sum_{i=1}^{\tau} \frac{\lambda_i^{total}}{\lambda^{total}} \cdot t_i \qquad \textit{and} \qquad E(B^2) = \sum_{i=1}^{\tau} \frac{2\cdot\lambda_i^{total}}{\lambda^{total}} \cdot t_i^{\,2}$$

Using the *Pollaczek-Khinchin formula* the mean waiting time at a database site can then be derived to be

[13] One could wonder at this point why the total number of messages in the *whole* system is used to calculate the moments of the service time distribution at a *single* site. Indeed, a single site only has to process $1/n$ of the total number of messages such that in the fractures above both the numerator and the denumerator should be divided by *n*, or equivalently multiplied by $1/n$. This factor however can be reduced from the fracture which in turn indicates that it is irrelevant to the *ratio* of the message types.

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

The resulting average response time for transactions of type *i* is

$$\overline{R_i} = \overline{W} + t_i + (1 - \ell_i) \cdot (\overline{W_c} + t_c^{send_i} + t_c^{return_i})$$

because a transactions 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.

F. Throughput

Since the utilization of a queueing system equals the product of arrival rate and mean service time ($\rho = \lambda \cdot E(B)$) the utilization ρ_D of a local database can be expressed as

$$\rho_D = \lambda^{total} \cdot \sum_{i=1}^{\tau} \frac{\lambda_i^{total}}{\lambda^{total}} \cdot t_i = \sum_{i=1}^{\tau} \lambda_i^{total} \cdot t_i = \sum_{i=1}^{\tau} \lambda_i \cdot \left(1 + (1 - q_i) \cdot (r_1 \cdot sel_i) \cdot (r_2 - 1) \cdot k\right) \cdot t_i .$$

Now the equation $\rho_D = 1$ can be solved for the arrival rate λ which yields the maximum database limited throughput T_D:

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

The utilization ρ_C of a local communication server is

$$\rho_C = \frac{\overline{N}}{n} \cdot E(B_c)$$

$$= \sum_{i=1}^{\tau} \left((1 - \ell_i) \cdot \lambda_i + (1 - q_i) \cdot (r_1 \cdot sel_i) \cdot (r_2 - 1) \cdot k \cdot \lambda_i\right) \cdot \left(t_c^{send_i}\right) + q_i \cdot (1 - \ell_i) \cdot \lambda_i \cdot t_c^{return_i}$$

so that solving $\rho_C = 1$ for λ results in the maximum communication limited throughput

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