Electronic Communications of the EASST
Volume 17 (2009)

Workshops der Wissenschaftlichen Konferenz Kommunikation in Verteilten Systemen 2009 (WowKiVS 2009)

Fuzzy Logic supported Consistency Management in DDVEs

Hermann Schloss, Ingo Scholtes, Jean Botev, Markus Esch, Alexander Höhfeld and Peter Sturm

12 pages

Guest Editors: M. Wagner, D. Hogrefe, K. Geihs, K. David
Managing Editors: Tiziana Margaria, Julia Padberg, Gabriele Taentzer
ECEASST Home Page: http://www.easst.org/eceasst/ ISSN 1863-2122

Fuzzy Logic supported Consistency Management in DDVEs

Hermann Schloss[1], Ingo Scholtes[1], Jean Botev[1], Markus Esch[2], Alexander Höhfeld[1] and Peter Sturm[1]

[1] System Software and Distributed Systems
University of Trier
54286 Trier, Germany
{schloss, scholtes, botev, hoehfeld, sturm}@syssoft.uni-trier.de

[2] Faculty of Sciences, Technology and Communication
University of Luxembourg
1359 Luxembourg, Luxembourg
markus.esch@uni.lu

Abstract: In distributed virtual environments, where avatars permanently change their properties (e.g. their position) or perform modifications on shared objects, inconsistent states may occur due to message latency or concurrent updates. Here, the consistency often falls prey to high interactivity and responsiveness demands. To minimize the presence of inconsistent states or data, the Elastic Consistency model aims at providing an optimal trade-off between consistency and system responsiveness. In this paper we present a Fuzzy Logic driven approach for the computation of this trade-off on the basis of environmental characteristics at the application runtime. Furthermore, we evaluate the impacts of this parameter on consistency and system responsiveness by adjusting the behavior of a classic mutual exclusion algorithm. In this paper we show that an acceptable degree of consistency with respect to required system responsiveness can be efficiently achieved, even when addressing a large number of users.

Keywords: Virtual Environments, Consistency, Fuzzy Logic

1 Introduction

Maintaining a consistent view of the virtual environment among different users is a key issue for the consistency provision in *Distributed Virtual Environments* (DVEs). The consistency problems which may occur in a DVE can be classified into *data consistency problems* arising from data replication and *shared state representation problems* resulting from the distribution of global state representations among all users within a considered *Area of Interest* (AoI). Data consistency problems arise mostly in round-based game scenarios, where for example several users concurrently access a "piece of treasure". Shared state representation problems are usually caused by latency delays in application scenarios with a high amount of interactivity between users resulting in different local views. Although our approach can be applied to both kinds of problems, in this paper we demonstrate the application of the Elastic Consistency model [SBE+08] to the handling of data items replicated among different users in DVEs.

Considering the fact that DVEs cannot be consistent and highly dynamic at the same time, we have to accept the presence of inconsistencies to some degree. Especially in multiuser *Decentralized Distributed Virtual Environments* (DDVEs), where consistency often falls prey to high user interactivity and system responsiveness demands, the presence of inconsistent states or data copies results in retroactive rollback events which negatively affect user experience.

To minimize the presence of inconsistent states or data, the Elastic Consistency model aims at providing an optimal trade-off between consistency and system responsiveness. For this purpose we compute a trade-off parameter value at application runtime on the basis of locally available environmental information or characteristics, and use this parameter to adjust the behavior of algorithms guaranteeing consistency of the shared state or replicated data. Environmental characteristics which can affect consistency provision in DDVEs are, for example, the number of users or replicated objects in an AoI, AoI's size, as well as system load and message transmission delays.

The central question which we address in this paper is, which correlation between different environmental characteristics does exist and which impact this characteristics have on the trade-off parameter value and consequently also on the consistency degree. This impact is easy to describe informally (e.g. with an increasing number of avatars in a region, the guaranteed consistency degree decreases), but an exact mathematical function describing this correlation is difficult to find. To address this problem, we propose a Fuzzy Logic driven approach for the computation of the trade-off parameter on the basis of environmental information at the application runtime.

In the course of this paper, we provide a brief overview of consistency aspects in virtual environments in section 2. Afterwards, we describe the determination of the trade-off parameter by means of Fuzzy Logic in section 3 and evaluate the impacts of this parameter to consistency and systems responsiveness in section 4. In section 5 we discuss alternative approaches for consistency handling in DDVEs. Finally, in section 6 we conclude with a brief overview of our contribution and future plans.

2 Consistency in Virtual Environments

In highly dynamic centralized (client-server based) DVEs where avatars permanently change their properties (e.g. their position) or perform modifications on shared objects, inconsistent states may occur due to message latency. The *Consistency-Throughput trade-off* introduced by [SZ99] states that a distributed virtual environment cannot be consistent and highly dynamic at the same time. One technique to guarantee consistency is to use an agreement protocol, where a user propagating an update of its state has to wait for acknowledgments from all other involved users before it can proceed. However, this technique increases the processing overhead and results in delays reducing throughput and affecting the responsiveness of the environment, which is not acceptable for a wide range of application scenarios in virtual worlds. To mitigate this problem in [Arm06] several latency compensation techniques improving user experience are described.

However, in decentralized (peer-to-peer based) DDVEs (e.g. HyperVerse [BEH+08]), these inconsistencies may additionally result from concurrent update propagations. The consistency problem here would not be recovered only by reducing the update throughput or by mitigating latency impacts by means of latency compensation techniques. An algorithmic handling is required to ensure the consistency of shared state or data items replicated among users in decentralized virtual environments.

To address this problem we define the term *Consistency-Responsiveness Trade-off* (CRT), which characterizes the contradicting objectives - system responsiveness and consistency provision - in the light of concurrent update propagations. The objective of our work is to find an optimal trade-off providing the highest possible consistency degree with respect to required system responsiveness. Since the use of traditional algorithms providing absolute consistency would increase processing time and negatively affect the system responsiveness, these algorithms have to be either modified (parameterized) or novel algorithmic approaches adjusting their behavior depending on application requirements (e.g., system responsiveness), system settings and characteristics have to be developed for consistency provision in DDVEs.

To handle the consistency problems in DDVEs in a flexible way, the Elastic Consistency model has been introduced in [SBE+08]. According to this model, application scenarios can be classified into three consistency levels, depending on their consistency and responsiveness requirements. Similar to the *consistency plug-ins* approach introduced by [SSW+08], at each consistency level - depending on the considered application scenario and addressed consistency problem (data consistency or state representation problem) - we assign a consistency plug-in that encapsulates a protocol implementing a specified consistency model. According to our notion, each algorithm used as consistency plug-in has to adjust its behavior depending on the CRT parameter value.

We propose the computation of the CRT parameter by means of Fuzzy Logic on the basis of heterogeneous environmental characteristics. Every time one of the considered characteristics changes, the parameter value is dynamically (at application runtime) adjusted. That is, with an increasing number of involved users or a decreasing message transmission delay, the value of the CRT parameter is recomputed and adjusted to the new situation. Consequently, all used algorithms have to adapt their behavior with respect to the new CRT value (low parameter value $\Rightarrow$ efficient, high parameter value $\Rightarrow$ consistent). Because of the linear runtime complexity necessary for the dynamic recomputation of the CRT value by means of Fuzzy Logic, the resulting overhead is negligible.

3 Computation of the trade-off parameter

Fuzzy Logic is known to provide a modeling and computation mechanism for *fuzzy* information - like *the number of users is large* - and to achieve provable valid and heuristically good solutions for a wide range of applications, especially in fields where an exact mathematical solution is difficult to find.

We demonstrate the application of Fuzzy Logic for the CRT parameter computation depending on settings, properties and characteristics of the virtual environment as well as depending on a concrete application scenario. For this purpose, we consider a simple scenario where objects located in an AoI are replicated among all avatars residing within this AoI.

In this simple scenario we assume that we have only two input variables - *number of users* and *number of objects* within an AoI - for CRT computation[1]. That is, to provide a graded Elastic Consistency in this scenario (completely neglecting system or network properties) we have to find a function

$$CRT : \mathbb{N} \times \mathbb{N} \rightarrow \{x \in \mathbb{R} \mid 0 \leq x \leq 1\}$$

computing the continuous trade-off parameter $0 \leq CRT \leq 1$ on the base of given user and object numbers.

The crux of the Fuzzy Logic is the definition of *fuzzy sets*. According to [Zad65], a fuzzy set is characterized by a membership function assigning to each element a grade of membership between zero and one. By this means, Fuzzy Logic is a generalization of the classic set theory where an element is either in a set (grade of membership = 1) or it is not (grade of membership = 0). However, this hard and discrete change of membership often contradicts with the human perception of continuously changing objects or properties. For example, considering the human perception of room temperature, there is no sharp definition of “cold” or “warm”. Zadeh’s Fuzzy Logic is a mathematical model for capturing human perception of things and states. It extends the definition of set operations such as *union* $\cup$ and *intersection* $\cap$ (sometimes called *and* and *or* respectively) and allows the application of these operations to fuzzy sets. By this means, Fuzzy Logic provides a mathematical formalism for handling linguistic state classifications. The inference scheme of a Fuzzy Logic system is shown in Figure 1. Hereby input variables are first

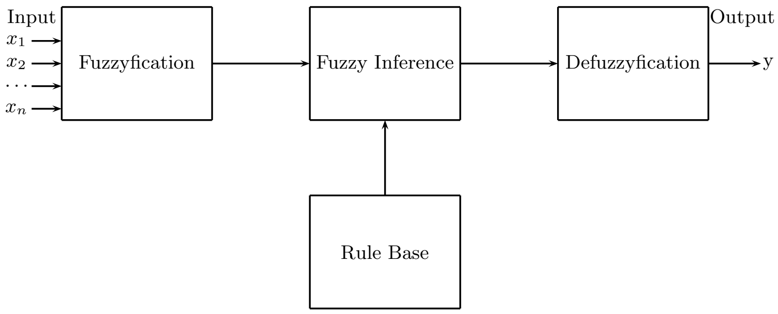


Figure 1: Fuzzy Inference

transformed into a vector of membership grades to fuzzy sets which are then used together with a rule base for the computation of a fuzzy output. Afterwards, this output has to be transformed into a “crisp value”. Further detailed information, definitions and proofs regarding the Fuzzy Logic can be found in [Zad65, KY96].

As mentioned above, in our sample scenario we consider two input parameters for the computation of the CRT parameter value: The number of involved users and the number of replicated object copies. According to the approach described in [Zad96], we define two *linguistic variables* NUMBER OF USERS and NUMBER OF COPIES. For the first linguistic variable we define the *linguistic values* (fuzzy sets): SMALL, MEDIUM, LARGE and VERY LARGE (Figure 2(a)). Analogously, the second variable can take the linguistic values SMALL, MEDIUM and LARGE (Figure 2(b)).

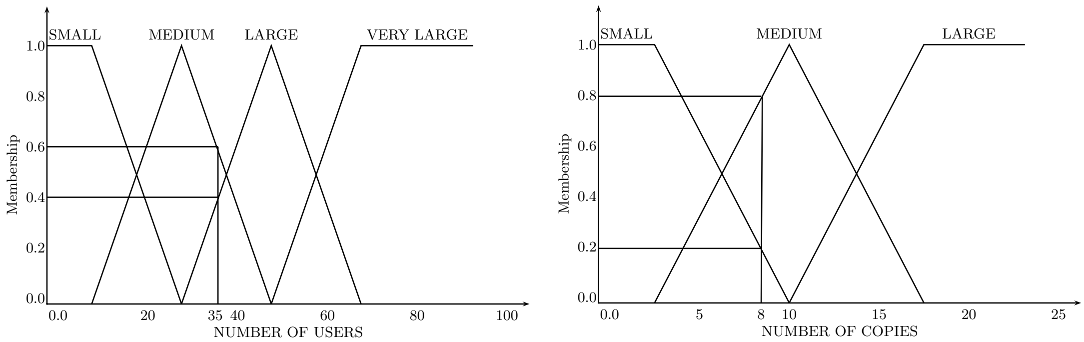
[1] In real world application scenarios further factors affecting the consistency in DVEs have to be considered.

We assume that there are 8 data items replicated among 35 users. According to the inference scheme depicted in Figure 1 these crisp values first have to pass the *fuzzyfication* stage where they are transformed into vectors of membership grades. That is, in the course of fuzzyfication a crisp value x_1 is mapped to the fuzzy set T^1 with degree μ^1, to the fuzzy set T^2 with degree μ^2 and so on.

Figure 2(a) shows the transformation of the exact value NUMBER OF USERS = 35. Hereby we get the following membership grades:

- $\mu_{SMALL}(35) = 0$
- $\mu_{MEDIUM}(35) = 0.6$
- $\mu_{LARGE}(35) = 0.4$
- $\mu_{VERYLARGE}(35) = 0$

The fuzzyfication of the number of replicated objects indicated by an exact value 8 is shown in Figure 2(b).



(a) Fuzzyfication of value 35

(b) Fuzzyfication of value 8

Figure 2: Fuzzyfication of Input Variables

Finally, we define the CRT parameter value as a linguistic variable with linguistic values SMALL, MEDIUM and LARGE 4(a). The continuous definition range of the parameter is $0 \leq CRT \leq 1$, where the value 1 implies that absolute consistency can be guaranteed. In opposition, the value 0 denotes that no consistency can be guaranteed. A value $0 < CRT < 1$ provides a trade-off between consistency and system responsiveness.

On the basis of defined fuzzy sets we define a rule base (Figure 3) which states the dependency of the (fuzzy) variable CRT VALUE from (fuzzy) variables NUMBER OF USERS and NUMBER OF COPIES for all consistency algorithms exhibiting a monotonically increasing runtime behavior, with an increasing number of users.

Considering our example, where 8 data items are replicated among 35 users and the corresponding membership grades shown in Figures 2(a) and 2(b) four rules from the rule base are firing:

1. when #USERS = MEDIUM and #COPIES = SMALL then CRT VALUE = LARGE

NUMBER OF COPIES \ NUMBER OF USERS	SMALL	MEDIUM	LARGE	VERY LARGE
LARGE	MEDIUM	SMALL	SMALL	SMALL
MEDIUM	LARGE	MEDIUM	SMALL	SMALL
SMALL	LARGE	LARGE	MEDIUM	SMALL

Figure 3: Rule base for linguistic CRT values

2. when #USERS = MEDIUM and #COPIES = MEDIUM then CRT VALUE = MEDIUM
3. when #USERS = LARGE and #COPIES = SMALL then CRT VALUE = MEDIUM
4. when #USERS = LARGE and #COPIES = MEDIUM then CRT VALUE = SMALL

In order to determine the firing strength α_i of i^{th} rule ($1 \leq i \leq 4$), we use the *min* function for the $\cap$ (*and*) operator[2].

$$\begin{aligned}
\alpha_1 &= \min\left(\mu^{\#USERS}_{MEDIUM}(35), \mu^{\#COPIES}_{SMALL}(8)\right) = \min(0.6, 0.2) = 0.2 \\
\alpha_2 &= \min\left(\mu^{\#USERS}_{MEDIUM}(35), \mu^{\#COPIES}_{MEDIUM}(8)\right) = \min(0.6, 0.8) = 0.6 \\
\alpha_3 &= \min\left(\mu^{\#USERS}_{LARGE}(35), \mu^{\#COPIES}_{SMALL}(8)\right) = \min(0.4, 0.2) = 0.2 \\
\alpha_4 &= \min\left(\mu^{\#USERS}_{LARGE}(35), \mu^{\#COPIES}_{MEDIUM}(8)\right) = \min(0.4, 0.8) = 0.4
\end{aligned}$$

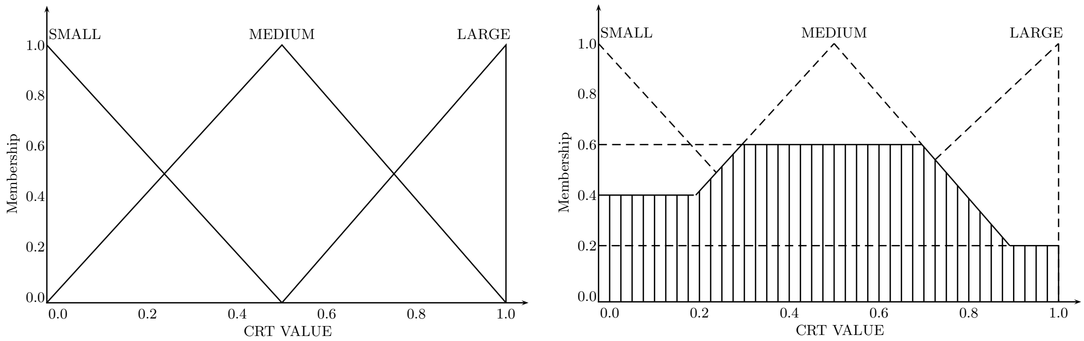
The firing strength of these rules is used to shape the corresponding output fuzzy set. That is, the output fuzzy set is restricted in its height by the firing strength value α_i of the corresponding rule. More precisely the membership μ of the output set y and for a rule i is defined as

$$\mu^i_y(w)' = \min(\alpha_i, \mu^i_y(w))$$

where w is a variable from the *support range* ($\mu > 0$) of the considered membership function.

The next step is the composition of the four output fuzzy sets into one single composite fuzzy set. This process is called *aggregation*. The output of the aggregation process is a single fuzzy set representing the output variable. If several rules map to the same output fuzzy set (see rules 2 and 3) in the most cases the maximum of the rule firing strength is taken for the composed output fuzzy set. By this means, in our example we will get the intended CRT value as a fuzzy set (Figure 4(b)).

[2] As alternative the multiplication product may be used instead



(a) Variable definition

(b) CRT value as a fuzzy set

Figure 4: Identification of the output variable CRT VALUE

The aggregated composite fuzzy set serves as input for the *defuzzification* process, which is defined as the process of mapping a fuzzy set to a crisp value σ. Several existing defuzzification techniques are evaluated in [Smi00]. In our approach we prefer the utilization of the *Center of Gravity* (CoG) method given by

$$\sigma_{CRT} = \frac{\int_0^1 y \cdot \mu_y(y)dy}{\int_0^1 \mu_y(y)dy} \approx 0.42$$

where y indicates the CRT VALUE axis.

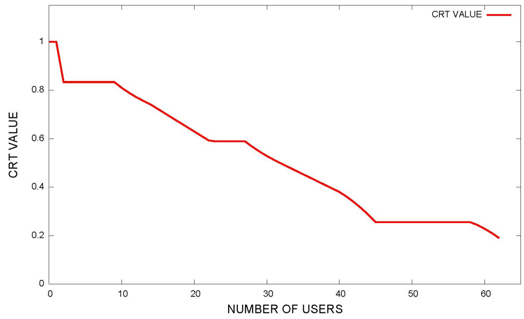
Figure 5 represents the behavior of the CRT parameter depending on a constant number of objects (8) and on a variable number of users (N). Hereby, Figure 5(a) considers the parameter values in the range $0 \le N \le 63$ and the Figure 5(b) in the range $0 \le N \le 1000$. As indicated by Figure 5(b) the CRT value for large number of users is constant. This is due to the fact, that for $N \ge 63$ the CoG value is not changing anymore.

In our example, the behavior of the CRT parameter value depends on the definition of the fuzzy sets representing the number of users and the number of objects. Such a definition necessitates a lot of manual configuration. To facilitate the definition of fuzzy sets, we look at some optimization schemes [Chi94] adapting the shape, support and tolerance of fuzzy sets with respect to the cluster estimation of the considered input variables.

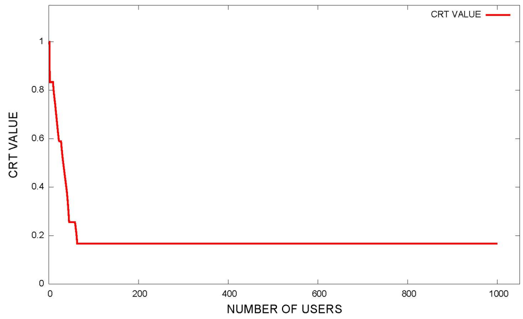
4 Evaluation of CRT Impacts

To clarify the impacts of the CRT parameter on consistency and system responsiveness, in this section we illustrate the general concept of the Elastic Consistency with a well-known example for the distributed mutual exclusion algorithm proposed by Maekawa [Mae85].

In the original Maekawa algorithm, a user wanting to enter the critical section first have to ask a certain so-called Voting Set of users and then to wait for their replies before entering the



(a) Up to 63 users



(b) Up to 1000 users

Figure 5: CRT Parameter Value

critical section. Users which are not willing to enter reply immediately. Users in the critical section queue the request and do not reply immediately. In the near-optimal case, the Voting Set comprises $\sqrt{N}$ users.

In realistic networks without Quality of Service guarantees, latencies can be highly variable. We now consider the original Maekawa's algorithm in such networks when no other user is currently in the critical section. In this case, it is clear that a single user in the Voting Set with a bad response time can massively influence the system responsiveness. With our notion of Elastic Consistency, we intend to relax this issue at least in situations where the sporadic violation of mutual exclusion safety can e.g. be coped with by means of rollback mechanisms.

For this, we first need to consider a slight variation in the original Maekawa's algorithm. We assume that each time, a reply is not immediately sent back to the requester, a negative reply is sent instead. Since we intend to wait only for the replies of the so called Reply Set (a fraction of the Voting Set) in order to mitigate the effect of users with bad response times, this is necessary to distinguish between users not responding because of the protocol and users not responding due to latency. Negative replies are not counted as original Maekawa's replies but are rather used to tell a user that it has to actually wait for a (normal, non-negative) reply because the critical section is occupied. In the following we study how the application of Elastic Consistency affects mutual exclusion safety (consistency) and responsiveness in this example.

Let $\overline{C}$ be an *inconsistency* event where u_1 enters into the critical section although another user u_2 is already within. By asking $\sqrt{N}$ users the probability for inconsistency is $P(\overline{C}) = 0$. However, as argued before considering a large number of users, waiting for replies of $\sqrt{N}$ users may negatively affect system responsiveness.

We assume that users utilize the CRT parameter by making the decision whether or not to enter after receiving only $CRT \cdot \sqrt{N}$ replies (Reply Set). The probability of $\overline{C}$ in this case is equal to the probability for getting $CRT \cdot \sqrt{N}$ non-negative replies, although one of the dangling $\sqrt{N} - (CRT \cdot \sqrt{N})$ replies is negative.

To compute the $P(\overline{C})$ we use the *hypergeometrical distribution* which is defined as a discrete probability distribution that describes the number of successes in a sequence of n draws from a finite population without replacement. We assume that $R = \sqrt{N}$ is the number of replies in the Voting Set expected by Maekawa's algorithm, M is the number of all existing non-negative replies, $n = CRT \cdot R$ is the number of replies in the Reply Set expected by our modified algorithm and m is the number of non-negative replies received by our modified algorithm. We compute the $P(\overline{C})$ as

$$P(\overline{C}) = \frac{\binom{M}{m} \cdot \binom{R-M}{n-m}}{\binom{R}{n}} \tag{1}$$

and define the term *consistency degree* as

$$P(C) = 1 - P(\overline{C}) \tag{2}$$

Applied to our sample scenario (35 users and 8 objects) a user u_1, which intends to enter into the critical section, has to make its decision after receiving $CRT \cdot \sqrt{N} = 0.42 \cdot \sqrt{35} \approx 3$ instead of 6 replies. Here, an inconsistent state $\overline{C}$ is given if u_1 receives 3 non-negative replies, but one of the remaining replies is negative. The probability of inconsistency $P(\overline{C})$ here is given by

$$P(\overline{C}) = \frac{\binom{5}{3} \cdot \binom{6-5}{3-3}}{\binom{6}{3}} = \frac{1}{2}$$

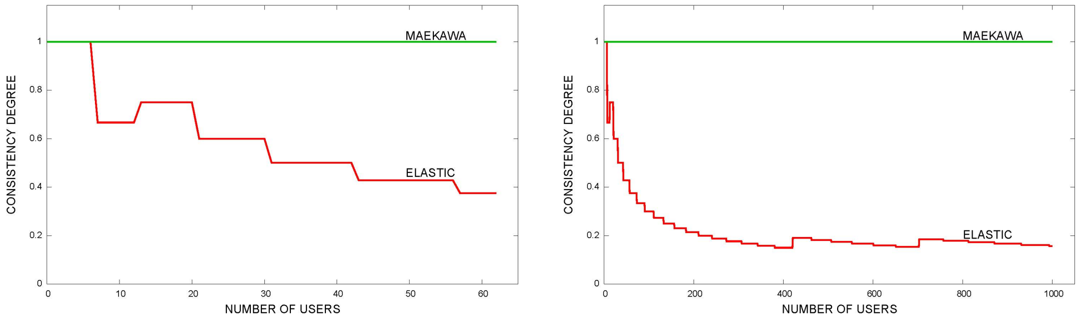
By this means the degree of consistency in this case is

$$P(C) = 1 - \frac{1}{2} = 0.5$$

To evaluate the impacts of the CRT parameter, we have evaluated the consistency degree according to the formulas 1 and 2 by considering up to 1000 users. Figure 6 indicates the consistency behavior of our modified algorithm in comparison to Maekawa's original. The guaranteed consistency degree that we can achieve by using our modification is acceptable considering a low number of users (Figure 6(a)). Even considering a large number of users we can guarantee consistency in about 20% of the concurrent accesses (Figure 6(b)). Due to formulas 1 and 2, each time a new user is added to the Reply Set (see Figure 7) the consistency degree temporarily increases (see Figure 6).

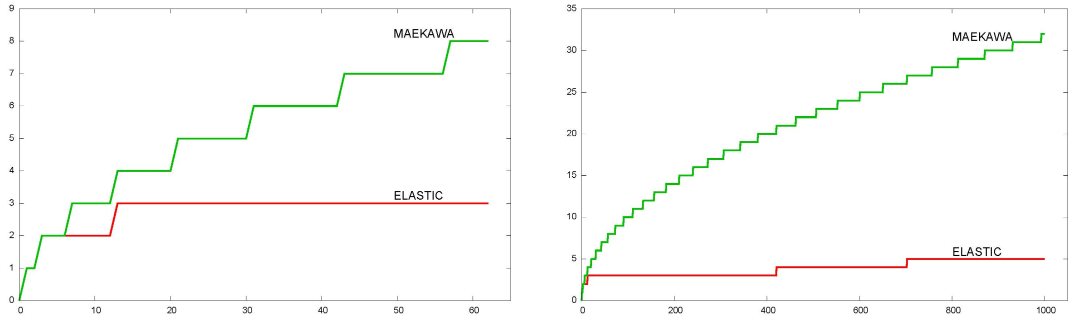
Figure 7 compares the number of replies expected by our elastic variation (Reply Set) and by Maekawa's original algorithm (Voting Set) with respect to a growing number of users. As indicated by this figure the number of replies expected by our algorithm is much lower. According to this fact, an improvement of system responsiveness can be assumed, since even a slight reduction of the number of the replies to wait for might be sufficient to exclude the users with the worst response times.

As shown in this section, by using the CRT parameter and adjusting the behavior of consistency algorithms it is possible to improve system responsiveness and to guarantee an acceptable degree of consistency.



(a) Up to 63 users (b) Up to 1000 users

Figure 6: Achieved Degree of Consistency



(a) Up to 63 users (b) Up to 1000 users

Figure 7: Expected Number of Replies

5 Related Work

In [YV00] the semantic space between traditional strong and optimistic consistency models for replicated services is evaluated. Similar to our notion of the Elastic Consistency, the authors argue that there is an important class of application scenarios which tolerate lower consistency degrees, but benefit from performance improvement. They present a metric-based middleware layer which enforces arbitrary consistency bounds among replicas and show a significant performance improvement when using their framework.

A consistency management infrastructure for consistency handling in *Massively Multiuser Virtual Environments* is introduced by [SSW+08]. Here, the authors propose the use of different plug-ins providing different degrees of consistency, which may be chosen depending on the application scenario. Another approach comprising three consistency levels addressing the consistency of the dynamic shared state in large-scale virtual environments is introduced by [LLL99]. In order to provide maximum performance, the approach switches automatically from one level to another, depending on number of users and system load. In our approach, depending on the considered application scenario and addressed consistency problem at each level, we assign a certain consistency algorithm and adjust its behavior dynamically depending on the CRT value.

In order to make a DVE as consistent as possible, [ZCLT04] introduce a time-space consistency metric based on environmental characteristics. The authors show how these characteristic parameters of a DVE are interrelated to each other and which impacts they have on the time-space inconsistency metric. By using this metric it is possible to determine the required degree of consistency without an actual execution of the DVE application, which is especially useful in the design stage of a new DVE. However in opposition to our approach this metric cannot be adjusted at application runtime.

In order to find optimal values for parameters influencing consistency and scalability in virtual environments in an ad hoc manner, [PAM07] use evolutionary optimization. Our Fuzzy Logic based approach however, provides much more comprehensible results.

6 Conclusion and Future Work

In DDVEs inconsistent states may occur due to message latency or concurrent updates. To minimize the presence of inconsistent states or data, the Elastic Consistency model - consisting of three consistency levels and assigning adjustable consistency plug-ins at each level - aims at providing an optimal trade-off between consistency and system responsiveness.

In the course of this paper we have presented a Fuzzy Logic driven approach for the computation of the CRT parameter value at the application runtime on the basis of locally available environmental characteristics. Depending on the CRT parameter value we can dynamically adjust the behavior of the consistency algorithms. As shown in this paper, by adjusting the behavior with respect to the CRT value, we can achieve acceptable system responsiveness and consistency degree.

The usage of traditional consistency algorithms (e.g. Maekawa's algorithm) as a basis for Elastic Consistency will most probably not be an option in real-world scenarios considering massive number of users. Therefore, in our future work we are going to address the development of novel probabilistic approaches for consistency handling in these scenarios.

Also the identification of further environmental characteristics, which can be considered for computation of the CRT parameter value, as well as the optimization of the fuzzy sets are medium-term goals.

Bibliography

[Arm06] G. Armitage. *Networking and Online Games*. John Wiley & Sons, 2006.

[BEH+08] J. Botev, M. Esch, A. Höhfeld, H. Schloss, I. Scholtes. The HyperVerse - Concepts for a Federated and Torrent-Based "3D Web". In *Proceedings of The 1st International Workshop on Massively Multiuser Virtual Environments at IEEE Virtual Reality 2008 (MMVE)*. 2008.

[Chi94] S. L. Chiu. Fuzzy Model Identification Based on Cluster Estimation. *Journal of Intelligent and Fuzzy Systems* 2(3):267–278, 1994.

[KY96] G. J. Klir, B. Yuan (eds.). *Fuzzy sets, fuzzy logic, and fuzzy systems: selected papers by Lotfi A. Zadeh*. World Scientific Publishing Co., Inc., River Edge, NJ, USA, 1996.

[LLL99] T. Lu, M.-T. Lin, C. Lee. Control Mechanism for Large-Scale Virtual Environments. *J. Vis. Lang. Comput.* 10(1):69–85, 1999.

[Mae85] M. Maekawa. A $\sqrt{N}$ algorithm for mutual exclusion in decentralized systems. *ACM Trans. Comput. Syst.* 3(2):145–159, 1985. doi:http://doi.acm.org/10.1145/214438.214445

[PAM07] S. Parkin, P. Andras, G. Morgan. Evolutionary Optimization of Parameters for Distributed Virtual Environments. In *IEEE Congress on Evolutionary Computation*. 2007.

[SBE+08] H. Schloss, J. Botev, M. Esch, A. Höhfeld, I. Scholtes, P. Sturm. Elastic Consistency in Decentralized Distributed Virtual Environments. In *Proceedings of the Fourth International Conference on Automated Solutions for Cross Media Content and Multichannel Distribution (AXMEDIS 2008)*. Pp. 249–253. Florence, Italy, 2008.

[Smi00] F. S. Smith. Selection of Fuzzication, Inference and Defuzzication Techniques for Fuzzy Logic Control. Technical report, National University of Ireland, Galway, 2000.

[SSW+08] G. Schiele, R. Sueselbeck, A. Wacker, T. Triebel, C. Becker. Consistency Management for Peer-To-Peer-based Massively Multiuser Virtual Environments. In *Proceedings of The 1st International Workshop on Massively Multiuser Virtual Environments at IEEE Virtual Reality 2008 (MMVE)*. 2008.

[SZ99] S. Singhal, M. Zyda. *Networked virtual environments: design and implementation*. ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 1999.

[YV00] H. Yu, A. Vahdat. Design and evaluation of a continuous consistency model for replicated services. In *OSDI'00: Proceedings of the 4th conference on Symposium on Operating System Design & Implementation*. Pp. 21–21. USENIX Association, Berkeley, CA, USA, 2000.

[Zad65] L. A. Zadeh. Fuzzy Sets. *Information and Control* 8:338–353, 1965.

[Zad96] L. A. Zadeh. On the analysis of large-scale systems, *Fuzzy sets, fuzzy logic, and fuzzy systems: selected papers by Lotfi A. Zadeh,* World Scientific Publishing Co., Inc. Pp. 195–209, 1996.

[ZCLT04] S. Zhou, W. Cai, B.-S. Lee, S. J. Turner. Time-space consistency in large-scale distributed virtual environments. *ACM Trans. Model. Comput. Simul.* 14(1):31–47, 2004.