How to combine strong availability with weak replication of objects?

Alice Bonhomme and Laurent Lefèvre
Inria Remap LHPC, LIGIM Bat. 710
Universit Claude Bernard Lyon1
69622 Villeurbanne Cedex, France
Email: {abonhomm,llefevre}@lhpca.univ-lyon1.fr

Paper for ECOOP'98 Workshop on Mobility and Replication

Abstract

Reducing objects replication without reducing objects availability is a critical point for DSM Systems. So far, most of the consistency models integrate either weak replication or strong availability but not both.
In this paper, we present a new consistency model that combines both, low replication and strong avalability. This model is based on the notion of group introduced by the DOSMOS system [1, 2].

keywords : DSM, consistency model, object replication

1 Introduction

Distributed Shared Memory (DSM) Systems are now well recognized to be an easy way to develop parallel applications. Now, one of the challenges of DSM Systems is to execute the applications in an acceptable execution time. For this purpose, choices of implementation have to be made, in terms of communication layer, memory consistency, data granularity...
In particular, one issue that DSM sytems have to address concerns minimising replication (ie reducing the number of messages required to keep the memory consistent) and maximising availability (ie increasing the number of local accesses) of shared data (objects). However, the existing consistency models integrate either weak replication or strong availability but rarely both of them [3].
New kinds of coherence are under study, Object consistency [4], Scope consistency [5], in order to limit the impact of memory coherence. While some systems try to limit objects replication by studying the cost of a replication [3], we present a new consistency model that tries to combine both aspects : strong availability and weak replication of objects.
This model is based on the notion of groups introduced in the object-based DSM System DOSMOS [1]. In this system groups are defined as sets of processes that share same objects. Thus, the replication of an object is reduced to the members of associated group. This provides strong availability of objects for their frequent users and no replication for infrequent accesses.

2 Motivations

In a DSM system, strong availability means that most of accesses to an object are local ones and consequently more performant than remote ones. On the other side, weak replication means a small number of copies of an object distributed among the nodes of the network. Thus the

[1]DOSMOS : Distributed Objects Shared MemOry System [1, 2]

management of the consistency of the copies is easier and generates few messages.
However, at first sight, one could easily think that it is impossible to combine both criteria in a system. Basically, a process is able to perform a local access to an object only if it has a local copy of this object. Thus, strong availability requires to have a large amount of objects copies. This enables the criterion of weak replication to be fulfilled.
Some cases in point are Sequential Consistency (an example of strong consistency) and Release Consistency [6]. Sequential consistency implements weak replication and consequently strong availability. On the other hand, Release Consistency implements strong availability by providing a local copy of an object to each process that access it. This leads to high replication.
To reduce the number of objects copies, Kai Li [7] introduces the well-known mecanism of copysets that memorize the id of processes that have a copy of some parts of the shared memory. Like this, useless copies (copies of objects that the process doesn't use) are avoided. Nevertheless, this concept doesn't make any difference between processes that frequently access an object and ones that access this object in an exceptionel way [2]. Thus a process can keep a copy of an object during all the execution even though it accessed this object just once. This process will receive all modifications concerning an unuseful object.
To address this problem, we propose a model in which only a subset of the processes have a copy of an object. The particularity of those subsets of processes is that they don't regroup the processes that access the object at least once (like in the lists of copies), they rather regroup the processes that frequently access an object. We develop this concept in the next section.

3 Group Consistency Model

The DOSMOS System is an object based DSM System [1, 2]. The policy of this system is to manage the consistency of an object only within the group of processes that frequently use this object. Thus DOSMOS introduces a hierarchical view of one's application processes by creating groups of processes that frequently share the same object.
The concept of hierarchical groups in DSM Systems has also been studied by Li and all [8]. They manage in two different ways the inter-group consistency (with Entry Consistency [9]) and the intra-group consistency (with Lazy Release Consistency [10]). However the degree of hierarchy is limited to 2. Our model propose a more general approach without limiting the degree of hierarchy and also without fixing the consistency implementation in each group.

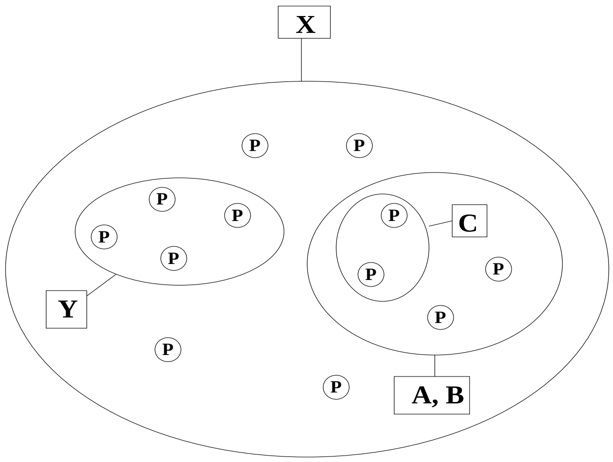

Figure 1: Groups of processes and objects

We define a group as a pair (G, O) where O is a set of objects and G a set of processes

sharing those objects. The figure 1 gives an example of 4 groups sharing the objects X, Y, A and B, C.
Furthermore, each group has a group manager (GMM) responsible for the communications inter-group. Each group management is independent from the others.

Concerning read and write operations to shared data, we classify the accesses into two categories : the intra-group accesses (the accessing process belongs to object's group) and the inter-group accesses between two distinct groups. Figure 2 shows an example of object accesses using group consistency. In this example, we have 2 sites A and B. In each site, processes share the same object (Y for site A and X for site B). The objects X and Y do not have the same consistency management : Release Consistency for Y and Lazy Release Consistency for X. In this context, we describe three differents actions : First, one process of site A writes Y, then this process sends the new value of Y to the other members of the site A. The second action illustrates an inter-group access. One process of site B wants to read the object Y. Since it does not belong to Y's group, it sends its request to its group manager that forwards it to the group manager of site A. This one forwards again the request to the right process in the group that sends the Y value to the requesting process . The last action concerns an acquire of X : since X is managed in Lazy Release Consistency, the acquirer asks the last acquirer for the new value of X.

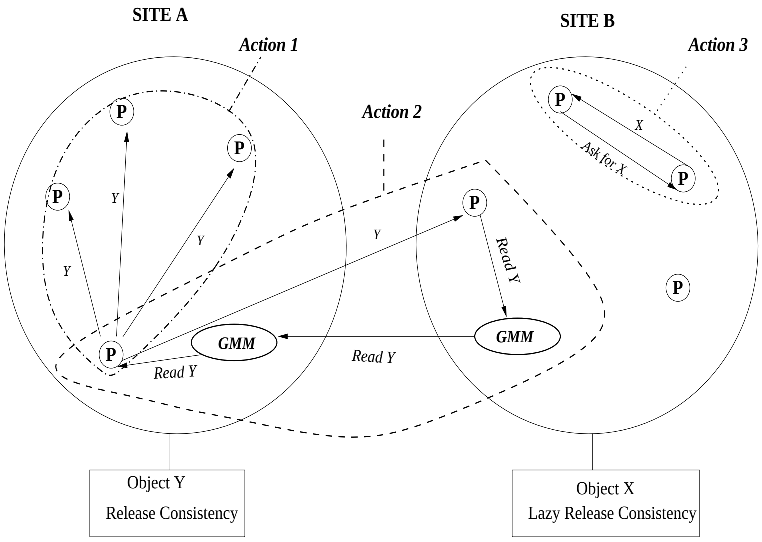

Figure 2: Three actions supporting the group consistency : an intra-write in Release Consistency (action 1), an inter-read (action2), and an acquire in Lazy Release Consistency (action 3)

For an intra-group access, we distinguish two cases :

- If it is the first access for the process to that object, the group manager sends it a copy of this object;
- Else, the process already has a local copy. Then, depending on the consistency implementation of the group, the process will directly work on this copy or not.

For an inter-group access, the accessing process doesn't have a copy and will never have one during the whole execution of the application. It simply asks for the value of this object (for a read) or sends the new value (for a write) to the group manager. But in any case, it gets a copy of this object.

In summary, in this model, we replicate objects only for the processes for which it is necessary. So we reduce replication without reducing availability.

4 Interest of the model and Applications

The notion of groups introduced in this model is an original approach for improving DSM Systems. It allows the system to be adapted to the environement under which it performs. By environment, we mean : type of processors, the communication network, kinf of applications, size of objects....
In fact, depending on the environement, the policy concerning replication of objects will not be the same and consequently the group structuration will be different.
In particular, group consistency is well suited for large scale systems. Systems made up of hundreds of workstations may require weak replication to garanty performances . Strong replication would effectively leads to a complicated management of the copies and thus a large exchange of messages. Furthermore, dynamic groups (cf 5) should easily permit load balacing, an important requirement for large Scale Systems.
Another case in point concerns the communication networks. This model fits perfectly the characteritics of various kind of networks. Depending on the latency, the bandwith or other criteria, we can design an accurate group structuration in order to reach the best compromise between replication and availability suited for the studied network.

Our model is designed to provide performances for large kind of applications. First, it is well-suited for MetaComputing and long distance cooperative applications . Thus, objects replication can be limited to the processes of the same site. Like this, we limit the communications between remote sites. Furthermore, if the different sites have an independant local network, we can desing a specification replication policy for each site.
Another application concerns the DataBases. The parallel DataBase Systems have to deal with objects replication and require at the same time fiability concerning the objects value and an acceptable respond time. Our model sounds to be well suited for the management of the data replication and eventually for load balacing for the requests.

5 Status of work and perspectives

We use the System DOSMOS to implement our model. This systems is an object-based DSM System and supports heterogeneity. In addition to the traditional primitives (acquire, release..) provided by most of the DSM Systems, it allows the user to build groups of processes and to choose the consistency implementation to use for each shared object.
Some experimentations to validate this model are under way. For those, we use different networks of workstations (Pentium under LINUX, Sparc, Alpha) connected with either Myrinet(we use the BIP Interface developped in our laboratory [11]), Ethernet or SCI networks.

Our model sounds sufficient for mostly regular applications. However, one could consider that this model is restricted by the fact that the groups are statically defined. Moreover, if we focus on irregular applications, dynamic groups seem to be a good alternative where a process can subscribe at runtime to a group and when it does not need it anymore, it just ends its subscribtion to the group. However, The management of dynamic groups in a load balancing perpective is an important part of our further work. We also continue our work on MetaComputing issues and especially on interactions between consistency model and communication networks.

References

[1] Laurent Lefevre. Parallel programming on top of DSM system. an experimental study. *Parallel Computing*, 23(1–2):235–249, April 1997.

[2] Lionel Brunie and Laurent Lefvre. New propositions to improve the efficiency and scalability of DSM systems. In IEEE, editor, *1996 IEEE Second International Conference on Algorithms & Architectures for Parallel Processing - ICA3PP '96*, pages 356–364, Singapore, June 1996.

[3] H.M Windisch. Improving the Efficiency of Objects Invocations by Dynamic Object Replication. In *Proc. of the International Conference on Parallel and Distributed Processing Techniques and Applications*, Athens(Georgia), 1995.

[4] C. K. Kazimierczak and B. Andersen. Object Consistency: A New Model for Distributed Memory Systems. In *Proc. of the Fourth Int'l Workshop on Object Orientation in Operating Systems (IWOOOS'95)*, pages 33–36, August 1995.

[5] L. Iftode, J. P. Singh, and K. Li. Scope consistency: A bridge between release consistency and entry consistency. In *Proc. of the 8th ACM Annual Symp. on Parallel Algorithms and Architectures (SPAA'96)*, pages 277–287, June 1996.

[6] S. V. Adve and K. Gharachorloo. Shared memory consistency models: A tutorial. *IEEE Computer*, 29(12):66–76, December 1996.

[7] Kai Li and Paul Hudak. Memory coherence in shared virtual memory systems. In *ACM Transactions on Computer Systems*, volume 7 (4), pages 321–359. ACM PRESS, November 1989.

[8] Q. Li, H. Ji, and L. Xie. Group consistency model which separates the intra-group consistency maintenance from the inter-group consistency maintenance in large scale DSM systems. *ACM Operating Systems Review*, 31(2):23–35, April 1997.

[9] Brian N. Bershad and Matthew J. Zekauskas. Midway: Shared memory parallel programming with entry consistency for distributed memory multiprocessors. Technical Report CMU-CS-91-170, Carnegie-Mellon University, Pittsburgh, PA (USA), September 1991.

[10] P. Keleher, A. L. Cox, and W. Zwaenepoel. Lazy release consistency for software distributed shared memory. In *Proc. of the 19th Annual Int'l Symp. on Computer Architecture (ISCA'92)*, pages 13–21, May 1992.

[11] Loc Prylli and Bernard Tourancheau. Bip : a new protocol designed for high performance networking on myrinet. In *Workshop PC-NO, IPPS/SPDP98*, Orlando, 1998.