Chapter 6
Improving Scalability in Replicated DRTDBS

Pratik Shrivastava
Madan Mohan Malaviya University of Technology, India

ABSTRACT

The demand for scalability in replicated distributed real-time database systems (RDRTDBS) is still explorative and, despite an increase in real-time applications, many challenges and issues remain in designing a more scalable system. The objective is to improve the scalability of the system during system scale up with new replica sites. Existing research has been mainly conducted in maintaining replica consistency between different replicas via replication protocol. However, very little research has been conducted towards improving scalability and maintaining mutual consistency and timeliness. Consequently, the ultimate aim of this chapter is to improve scalability in RDRTDBS such that performance of the system does not degrade even though new replica sites are added.

INTRODUCTION

In the current scenario, data is playing a prominent role in many applications and the back-end database system of these application stores such valuable data. This database system requires an efficient and effective management technique (Berrington J., 2007) such that database operation in terms of insertion, deletion, updation, and searching can be processed more easily (Ramakrishnan, R. & Gehrke J., 2000; Ullman, J. D., 1984). Additionally, these database operations can be processed in the centralized manner or distributed manner. The decision to process database operations in the centralized location or distributed location will depend upon the client's requirement (Garcia-Molina H. & Lindsay B., 1990; Bernstein, P. A. et al., 1987). If the application is designed for a small scale, then it is advisable to process database operation in a centralized location. However, if the application is on a large scale, then it is advisable to process database operation in a distributed fashion. Despite this, there are some other factors also that need to be considered such as the number of requests, number of system resources, and so on to quickly decide about the processing location for data operations.

DOI: 10.4018/978-1-7998-2491-6.ch006

Real time database system (RTDBS) in an emerging area of this database system which is specifically designed to work with real time systems (Aldarmi S. A., 1998) such as space shuttle, flight control system and some others. This database system has a stringent requirement to maintain the timeliness and temporal consistency of the admitted real time transaction (RTT) such that the system can be saved from catastrophic failure (Ulusoy Ö., 1995). As this database system continues to evolve, RTDBS is used in a large number of real time applications and is generating massive amounts of data. Distributed real time database system (DRTDBS) is an extension of RTDBS which is specifically designed to work with such a huge amount of data in a timely manner. The primary objective of DRTDBS is to satisfy the timeliness of the admitted RTT and also to maintain the temporal consistency of real time data items. In the past, large number of researches are conducted in different directions of the DRTDBS such that the performance of the system gets increased. These research directions include concurrency control protocol, buffer management, scheduling RTTs, commit protocol, replication technique and so on (Shanker U. et al., 2008).

Replication technique (RT) is the most focused area of research for the researcher in the DRTDBS. In this technique, data replicas are placed in different database sites such that admitted RTT can be processed locally. This extension of DRTDBS with RT is termed as RDRTDBS. Data replicas in DRTDBS is either fully, partially, or virtually fully replicated (Mathiason, G. et al., 2007, August). In Fully RDRTDBS, every admitted RTT is processed locally because, in the fully replicated scheme, all the nodes hold the same data copy. Thus, meeting the RTT deadline in RDRTDBS becomes easy. This replication scheme is suitable for a read intensive environment because read RTT uses a shared lock. This lock allows a greater number of read RTT to work on the same data item concurrently. However, a fully replicated scheme suffers from unnecessary bandwidth utilization, wastage of system resources, and unnecessary replica site updation. Partially replication scheme solves such identified issues more easily because, in partial replication, replicated sites are partially replicated and unnecessary bandwidth utilization gets overcome. The decision of placing the data replica in the partially replicated environment will depend on the number of requests and from the location of requests. This type of replication scheme is suitable for an update intensive environment where the majority of RTTs are of update in nature. Virtual full replication solves the issue of both the replication types more effectively. Thus, this replication type is more suitable for update and read an intensive environment. Apart from the replication type, the replica concurrency control protocol (RCCP)/replication protocol (RP)/replica update technique (RUT) plays a major role in the replication technique. RCCP processes to maintain replica consistency in between replicated sites such that consistent value can be provided to the admitted RTT. Although, majority of researchers have conducted research towards RCCP (Xiong M. et al., 2002; Son S. H. & Kouloumbis S., 1993; Peddi P. & DiPippo L. C., 2002; Haj Said A. et al., 2008; Shrivastava P. & Shanker U., 2018, August; Shrivastava P. & Shanker U., 2018; Syberfeldt S., 2007; Gustavsson S. et al.,2004; Kim Y. K., 1996; Mathiason G. et al., 2007; Son S. H. & Zhang F., 1995, April; Shrivastava P. & Shanker U., 2019, January; Son S. H. et al., 1996; Salem R. et al., 2016; Shrivastava P. & Shanker U., 2018, December; Gustavsson S. et al. 2005, April), research towards scalability, security (Shrivastava P. & Shanker U., 2020), dependency preservation, fault tolerance etc. are still in very infancy stage (Shrivastava P. & Shanker U., 2018 August). Thus, in the current chapter, our objective is to improve the scalability of the RDRTDBS such that more replica sites can be added without degrading the performance of the system.

In recently published article (Shrivastava P. & Shanker U., 2018), the authors have proposed a system model that maintains the replica consistency via the middleware. This system model consists of the master site to process only write and update RTT, slave site to process only read RTT and middleware