An improved commutative replicated data type for peer to peer collaborative editing

Xiao Lv [1, a], Yongjie Li[2,b] and Chaoqun Ang[3]

[1]College of Computer Engineering,Naval University of Engineering,Wuhan, China

[2]College of Computer Engineering,Naval University of Engineering,Wuhan, China

[3]College of Computer Engineering,Naval University of Engineering,Wuhan, China

[a]lvxiaoluxu@126.com, [b]cumtlyj@163.com

Keywords: collaborative editing; peer to peer network; commutative repalicated data type

Abstract. With the development of Internet, collaborative editing over Internert becomes a common office affair,which can support a harmonious human to human interaction.Peer-to-peer (P2P) networks are very efficient for distributed collabotative editing. Existing many collaborative editing algorithms depend on the number of sites, which cannot scale and suitable for P2P networks. In this paper, we propose a optimal collaborative editing algorithm, which can be addressed two challenges as follows: it is based on RGA(Replicated Growable Array) algorithm, which has been proved correctness. Secondly, it does not depend on the number of sites and can support peer to peer collaborative editing.

Introduction

Different from traditional distributed systems, real-time collaborative editing systems support many users from different places to deal with the same data simultaneously[1,2], which can bring a great chanllenge for the consistency maintenace in the process of collaborative editing.

Operational transformation (OT) algorithms as an optimistic consistency control method have been proposed nearly over the past three decades[3-8]. Local operations are always executed immediately when they are issued by the user. Remote operations need to be transformed with concurrent operations before execution to repair inconsistencies. As a result, they can achieve consistency of shared data at cooperating sites.In order to achieve consistency, OT mainly have two methods : one is to design a total order operational transformation path,which can be capatable of avoiding TP1/TP2; another is to design the operational transformation functions which can be capable of preserving TP1/TP. However, exsiting researches show that the second mechanism is very difficult to achieve, the first method is always to ensure the convergence[7,8].

The main problem of the operational transformation approach is scalability. Most existing algorithms such as GOT[9], GOTO[10] and ABT family algorithms[11,12] depend on state vectors to detect causal and concurrent operations. A state vector is composed of logical clocks of all sites, which need to kown the number of sites in collaboration[13]. Users cannot join or leave dynamically in the collaborative session. When the number of sites grows, the size of the statevectors is unbounded. Also, the time of detecting causal and concurrent operations will decline as state vectors grow. OT algorithms based on state vectors cannot scale and cannot be suitable for P2P networks.

Recently, a class of new method called CRDT(commutative repalicated data type) has been proposed[14-18].Concurrent operaions are designed to be commutative by using the characteristics of abstract data types. By associated each object with an unique and totallly ordered identifier, all objects can be totally stored in the data types and can ensure convergence for all sites. Experiments results show that CRDT algorithms outperform OT algorithms by a factor between 25 and 1000[19]. Also, RGA algorithm has been accepted to have the best average performance in typical CRDT algorithms.But in RGA algorithm, state vectors are used to define identifier for every object.

It is not suitable for P2P collaborative editing.

In this paper, we improve RGA algorithm, propose an optimized RGA algorithm(ORGA), which can be applied to P2P collaborative editing.The remainder of this paper is organised as follows:

Section 2 describes the RGA approach and details the existing problem. Section 3 presents the related definitions of ORGA algorithm and an instance analysis. Section 4 summarises the contributions of this paper and presents some perspectives.

RGA Algorithm

1) RGA control procedure

RGA can support insertion, deletion and update[16]. A linked list and a hash table are repliacated at all collaborative sites. A linked list stores all total ordered objects. A hash table reserves the pointer to the node in the linked list. The data structue is shown in Fig. 1.

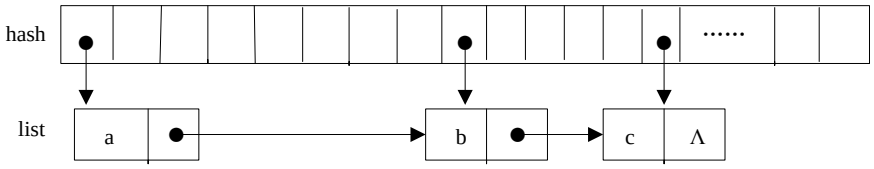

Fig.1. hash table and linked list in RGA

Every operation object has an s4vector including four integers,which is defined as <int ssn, int sid, int sum, int seq>.An s4vector can be used as a unique index and used to resolve conflits between concurren operation.Especially, ssn is a global session number that increases automatic, sid is the site ID unique to the site, sum is the sum of the state vector, seq is reserved for purging tombstones .

When users issue local operations, find their target elements of local operaion in the linked list, whcih can be achieved by integer indexes.If the operation is insertion, link the object after the target element in the linked list. If the operation is deletion, make the object tombstone in the linked list. Meanwhile, remote operations retrieve their target elements via the hash table with their unique s4vector. A remote insertion needs to compare its s4vector to other concurrent insertion objects's 4vector at the same position, then insert its object in a proper position. A remote deletion make the target element tomstone.

2) Existing problems

An s4vector is defined as <int ssn, int sid, int sum, int seq>. For example, Fig.2. shows a collaborative editing scenario,The state vector of all operations is SV_{O_1} =(1,0,0), SV_{O_2} =(0,1,0), SV_{O_3} =(0,0,1), SV_{O4}=(2,1,1).we need to kown the fixed number of sites in collaboration, then assign the state vectors for all operations.

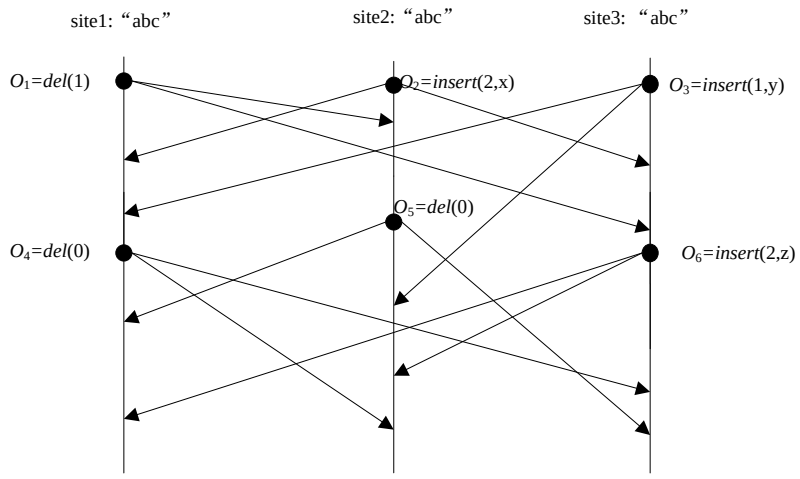

Fig.2. A collaborative edting scenario

An s4vector in RGA algorithm is unique, because s4vector order is a total order. Given two s4vectors, first the number of ssn is compared, then the sum of state vectors is compared if ssn is equal. At last, compare the number of sites if ssn is euqal and sum is equal. In Fig.2, the s4vectors of

all operations are s4vector (O_1) = < 1,1,1,1>, s4vector(O_2)=<1,2,1,0>,s4vector(O_3)=<1,3,1,0>, s4vector(O_4)=<1,1,4,2>.The s4vector total order of all operations is as follows: s4vector(O_1)$\prec$ s4vector(O_2)$\prec$ s4vector(O_3)$\prec$ s4vector(O_4).

From the above example , if we use state vectors to define s4vectors of all operations, we need to kown the number of all collaborative sites.Therefore, users cannot join or leave dynamically in the collaborative session. So it cannot scale and cannot be suitable for P2P networks.

ORGA Algorithm

1) The related definitions of ORGA

The control procedure of ORGA algorithm is the same as RGA algorithm. The execution functions of local insertion and local deletion are defined as Fig.3.

```
LocalInsert(ORGAOperation op)
    {
      create a new ORGA_Node;
      if (op.pos == 0)
      {Insert the new ORGA_Node after head;}
      else
      {   find the target Node;
          Insert the new ORGA_Node after the target Node; }
     hash.Add(ORGA_Node.key, ORGA_Node);
     }

LocalDelete(ORGAOperation op)
 {
   Find the target Node based on op.pos;
   Node.makeTombstone();
  }
```

Fig.3. The functions of local insertion and deletion

The execution functions of remote insertion and deletion are defined as Fig.4.

```
RemoteInsert(ORGAOperation op)
 {  create a new ORGA_Node;
    ORGA_Node prev, next;
    ORGAs4Vector s4v = op.gets4vector();
     if (op.getPos()=null) prev = head;
     else prev = (ORGA_Node)hash[op.getPos()];
     next = prev.getNext();
     while (next != null)
     { if (Next.getKey()  s4v) break;≺
        prev = next;
        next = next.getNext();
     }
     ORGA_Node.setNext(next);
     prev.setNext(ORGA_Node);
     hash.Add(op.gets4vector(),ORGA_Node);
  }

RemoteDelete(ORGAOperation op)
 {
    ORGA_Node node = (ORGANode)hash[op.getPos()];
    node.makeTombstone();
 }
```

Fig.4. The functions of Remote insertion and deletion

Different from RGA algorithm, ORGA algorithm does not use state vectors to define s4vector.

The s4vector of ORGA algorithm is defined as follows:

Definition 1. *s4vector*' is defined as <ssn, <sid, lc>, seq> .ssn,sid and seq are the same as RGA, lc is the logical clock per site, which is a global number that increases automatic.

Definition 2.*s4vector*'$\prec$.Given two operatons O_1 and O_2, the *s4vector*'of O_1 *is s4vector*'(O_1) and the *s4vector*'of O_2 *is s4vector*'(O_2). *s4vector*'(O_1) $\prec$ *s4vector*'(O_2), iff (1) the ssn of *s4vector*'(O_1) <the ssn of *s4vector*'(O_2); (2) the sid of *s4vector*'(O_1) <the sid of *s4vector*'(O_2) if

the ssn of *s4vector*'(O_1) =the ssn of *s4vector*'(O_2); (3) the lc of *s4vector*'(O_1) <the lc of *s4vector*'(O_2) if the ssn of *s4vector*'(O_1) =the ssn of *s4vector*'(O_2) and the sid of *s4vector*'(O_1) =the sid of *s4vector*'(O_2) .

In Fig.2, the *s4vectors*' of all operations are as follows: *s4vector*'(O_1) =<1,<1,1>,1>, *s4vector*'(O_2) =<1,<2,1>,0>, *s4vector*'(O_3) =<1,<3,1>,0>,*s4vector*'(O_4)=<1,<1,2>,2>. The *s4vector*' $\prec$ of all operations is as : *s4vector*'(O_1) $\prec$ *s4vector*'(O_4) $\prec$ *s4vector*'(O_2) $\prec$ *s4vector*'(O_3).

2) An instance analysis of ORGA

We take a specific collaborative editing scenario in Fig.2 as an example, then give all the steps of opearations execution at all sites and the consistent result theoretically.

Fig.2. shows a collaborative scenario, suppose three sites from the same initial state "abc". Three sites concurrently generates O_1, O_2, O_3 respectively. s_j^i represents the jth state of the ith site. α represents tombstone of the character α . The inital structure of hash and linked list is shown as Fig.5.

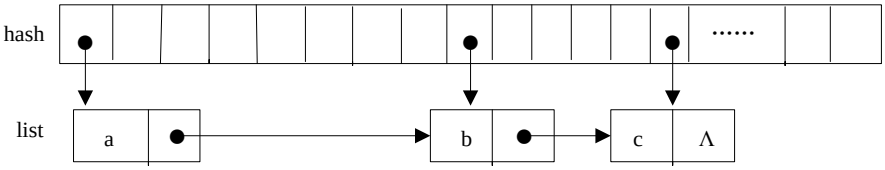

Fig.5. The initial structure of hash and linked list

At site1, execute O_1, set "b" as tombstone, s_1^1="a$\not b$ c". When receive O_2, execute O_2, s_2^1="a$\not b$ xc". When receive O_3, execute O_3, s_3^1="ay$\not b$ xc". When receive O_4, execute O_4, s_4^1="$\not a$ y$\not b$ xc".

At site2, execute O_2, s_1^2="abxc". When receive O_1, execute O_1, s_2^2="a$\not b$ xc". When receive O_3, execute O_3, s_3^2="ay$\not b$ xc".When receive O_4, execute O_4, s_4^2="$\not a$ y$\not b$ xc".

At site3, execute O_3, s_1^3="aybc". When receive O_2, execute O_2,="aybxc".When receive O_1, execute O_1, s_3^3="ay$\not b$ xc".When receive O_4, execute O_4, s_4^3="$\not a$ y$\not b$ xc".

At last ,all three sites get the same result "$\not a$ y$\not b$ xc" .

The final structure of hash and linked list is shown as Fig.6

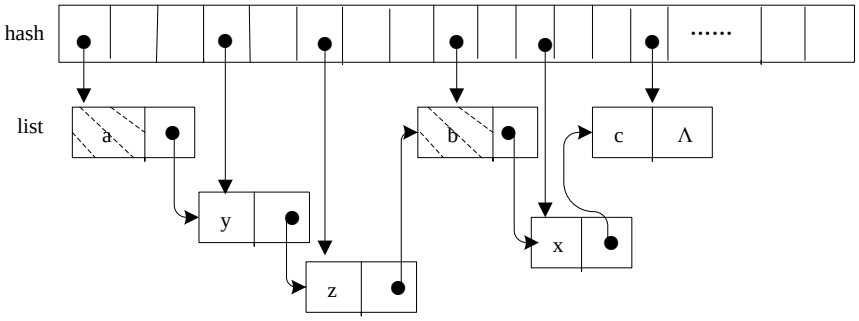

Fig.6 The final structure of hash and linked list

Summary

In this paper, we propose a optimized RGA algorithm called ORGA. In the ORGA algorithm, we use logical clocks to define the unique identifier instead of state vectors of RGA , which does not depend on the number of sites and suitable for peer to peer collaborative editing.

In future research, we plan to extend this work specifically for application domains such as collaborative software development , CAD and other complex collaborative systems.

References

[1] Ellis C A, Gibbs S J. *Concurrency control in groupware systems*. Proceedings of ACM SIGMOD international conference on Management of data. Portland, Oregon, USA, 18(2): 399-407(1989)

[2] Saito Yasushi, Shapiro Marc. *Optimistic replication*. ACM Computing Surveys, 37(1): 42-81.(2005)

[3] Ressel M, Nitsche-Ruhland D, Gunzenhäuser R.*An integrating, transformation-oriented approach to concurrency control and undo in group editors*. Proceedings of the ACM conference on Computer supported cooperative work. Boston, MA, USA. (1996)

[4] Prakash A, Knister M J. *A framework for undoing actions in collaborative systems*. ACM Transactions on Computer-Human Interaction (TOCHI), 1(4): 295-330.(1994)

[5] Sun Cheng-Zheng, Ellis C A. *Operational transformation in real-time group editors: issues, algorithms, and achievements*. Proceedings of the ACM conference on Computer Supported Cooperative Work.Seattle, WA, USA.(1998)

[6] Vidot N, Cart M, Ferrié J, Suleiman M. *Copies convergence in a distributed real-time collaborative environment*. Proceedings of the ACM conference on Computer Supported Cooperative Work. Philadelphia, PA, USA.(2000)

[7] Imine A, Molli P, Oster G, Rusinowitch M. *Proving correctness of transformation functions in real-time groupware*. Proceedings of the 8th European Conference of Computer-supported Cooperative Work.Helsinki, Finland.(2003)

[8] Sun C, Xu Y, Agustina A.*Exhaustive search of puzzles in operational transformation*. Proceedings of the 17th ACM conference on Computer supported cooperative work & social computing. Baltimore, MD, USA.(2014)

[9] Sun Cheng-Zheng, Zhang Yan-Chun, Jia Xiao-Hua, Yang Yun. *A generic operation transformation scheme for consistency maintenance in real-time cooperative editing systems.* Proceedings of the ACM conference on Supporting group work. Phoenix, Arizona, USA.(1997)

[10] Sun Cheng-Zheng, Ellis C A. O*perational transformation in real-time group editors: issues, algorithms, and achievements.* Proceedings of the ACM conference on Computer Supported Cooperative Work. Seattle, WA, USA.(1998)

[11] Li Du, Li Rui. *An admissibility-based operational transformation framework for collaborative editing systems*.Proceedings of the ACM conference on Comput Supported Cooperative Work. Savannah, Georgia, USA .(2010)

[12] Shao Bin, Li Du, Gu Ning. *ABTS: A transformation-based consistency control algorithm for wide-area collaborative applications*.Proceedings of the 5th International Conference on Collaborative Computing: Networking, Applications and Worksharing. Washington, DC,USA.(2009)

[13] Mattern F. *Virtual time and global states of distributed systems*. Parallel and Distributed Algorithms, 1(23): 215-226(1989)

[14] Preguica N, Marques J M, Shapiro M, Letia M. *A commutative replicated data type for cooperative editing*.Proceedings of the 29th IEEE International Conference on Distributed Computing Systems. Montreal, QC,Canada.(2009)

[15] Shapiro M, Preguica N. *Designing a commutative replicated data Type*. France,Inria, Report:

RR-6320(2007)

[16] Wu Q, Pu C, Ferreira J E. *A partial persistent data structure to support consistency in real-time collaborative editing*. Proceedings of 26th IEEE International Conference on Data Engineering (ICDE). Long Beach, CA,USA.(2010)

[17] Wu Q, Pu C. *Consistency in real-time collaborative editing systems based on partial persistent sequences*.Georgia:Georgia Institute of Technology, Report: GIT-CERCS-09-07.(2009)

[18] Roh H G, Jeon M, Kim J S, Lee J. Replicated abstract data types Building blocks for collaborative applications.Journal of Parallel and Distributed Computing,71(3): 354-368(2011)

[19] Ahmed-Nacer M, Ignat C L, Oster G, Roh H G. *Evaluating crdts for real-time document editing.*Proceedings of the 11th ACM symposium on Document engineering. Mountain View California,USA.(2011)