www.ijiarec.com

ISSN: 2348-2079

MAR-2015

International Journal of Intellectual Advancements and Research in Engineering Computations

REPLICA ALLOCATION OVER A MOBILE AD HOC NETWORK BY HANDLING SELFISHNESS

*Mr.S.Mohan Prasad

ABSTRACT

In a mobile ad hoc network, the mobility and resource constraints of mobile nodes may lead to network partition or performance degradation. Several data replication techniques have been proposed to minimize presentation degradation. Most of them assume that all mobile nodes work together fully in terms of sharing their memory space. In reality, however, some nodes may uncaringly decide only to cooperate partially, or not at all, with other nodes. These selfish nodes could then reduce the overall data accessibility in the network. In this paper, we examine the collision of selfish nodes in a mobile ad hoc network from the perspective of replica allotment. We term this selfish replica allocation. In particular, we develop a selfish node finding algorithm that considers partial selfishness and novel replica allocation technique to properly cope with selfish replica allocation. The conducted simulations demonstrate the proposed approach outperforms conventional cooperative replica allocation techniques in terms of data accessibility, statement cost, and average query delay.

Index Terms— Mobile ad hoc networks, degree of selfishness, selfish replica portion.

INTRODUCTION

MOBILE ad hoc networks (MANETs) have attracted a lot of attention due to the popularity of mobile devices and the advances in wireless communication technologies. A MANET is a peer-to-peer multihop mobile wireless network that has neither a fixed infrastructure nor a central server. Each node in a MANET acts as a router and communicates with each other. A large variety of MANET applications have been developed a MANET can be used in special situations, where installing infrastructure may be difficult, or even infeasible, such as a battlefield or a disaster area. A mobile peer-to-peer file sharing system is another interesting MANET application. Network partitions can occur frequently, since nodes. Move freely in a MANET, causing some data to be often inaccessible to some of the nodes. Hence, data accessibility is often an important performance metric in a MANET Data are usually replicated at nodes, other than the original owners, to increase data accessibility to cope with frequent network partitions. A considerable amount of research has recently been proposed for replica allocation in a MANET. However, this will increase its own query delay. A node may act selfishly, i.e., using its limited resource only for its own benefit, since each node in a MANET has resource constraints, such as battery and storage limitations. Existing research on selfish behaviors in a MANET mostly focus on network issues storage for the benefit of others. The number of selfish users has increased to 85 percent of all Gnutella users over five years. In this paper, we shall refer to such a problem as the selfish replica allocation. Simply, selfish replica allocation refers to a node's non cooperative action, such that the node refuses to cooperate fully in sharing its memory space with other nodes Accessibility. Let us consider the case where N3 behaves "selfishly" by maintaining M03 We believe that the partially selfish should also be taken into account, in addition to the fully selfish nodes to properly handle the selfish replica allocation problem. We therefore need to measure the "degree of selfishness" to appropriately handle the partially selfish nodes. Motivated by this concept of

Author for Correspondence:
*Mr.S.Mohan Prasad Assistant Professor, Department of IT SVCET, Puliyangudi, India, E-Mail id: mohanprasadme57@gmail.com

"partial selfishness," we borrow the notion of credit risk (CR) from economics to detect selfish nodes. Since the credit risk is calculated from several selfishness features in this paper, it can measure the degree of selfishness elaborately. In our scheme, a node can measure the degree of selfishness of another node, to which it is connected by one or multiple hops in a MANET. We devise novel replica allocation techniques with the developed selfish node detection method.

PRELIMINARIES

SYSTEM MODEL

In this paper, we assume that each node has limited local memory space and acts as a data provider of several data items and a data consumer. Each node holds replicas of data items, and maintains the replicas in local memory space. The replicas are relocated in a specific period. Any node freely joins and organizes an open MANET. We model a MANET in an undirected graph there are m nodes, N1; N2; . . .; Nm and no central server determines the access frequency does not change. Each node moves freely within the maximum velocity.

When a node Ni makes an access request to a data item, it checks its own memory space first. The request is successful when Ni holds the original or replica of the data item in its local memory. If it does not hold the original or replica, the request will be broadcast. The request is also successful when Ni receives any reply from at least one node connected to Ni with one hop or multiple hops, which holds the original or replica of the targeted data item. Otherwise, the request, or query processing, fails.

NODE BEHAVIOR MODEL

The work considers only binary behavioral states for selfish nodes from the network routing perspective: selfish or not. As mentioned in Section 1, it is necessary to further consider the partial selfish behavior to handle the selfish replica allocation. Therefore, we define three types of behavioral states for nodes from the viewpoint of selfish replica allocation2:

Type-1 node:

The nodes are non selfish nodes. The nodes hold replicas allocated by other nodes within the limits of their memory space.

Type-2 node:

The nodes are fully selfish nodes. The nodes do not hold replicas allocated by other nodes, but allocate replicas to other nodes for their accessibility.

Type-3 node:

The nodes are partially selfish nodes. The nodes use their memory space partially for allocated replicas by other nodes. Their memory space may be divided logically into two parts: selfish and public area. These nodes allocate replicas to other nodes for their accessibility.

PROPOSED STRATEGY

OVERVIEW

Each node detects the selfish nodes based on credit risk scores. Each node makes its own (partial) topology graph and builds its own SCF-tree by excluding selfish nodes. Based on SCF-tree, each node allocates replica in a fully distributed manner. The CR score is updated accordingly during the query processing phase. We borrow the notion of credit risk from economics to effectively measure the "degree of selfishness." In economics, credit risk is the measured risk of loss due to a debtor's nonpayment of a loan. A bank examines the credit risk of an applicant prior to approving the loan.

BUILDING SCF-TREE

The SCF-tree based replica allocation techniques are inspired by human friendship management in the real world, where each person makes his/her own friends forming a web and manages friendship by himself/herself. He/she does not have to discuss these with others to maintain the friendship. The decision is solely at his/her discretion. The main objective of our novel replica allocation techniques is to reduce traffic overhead, while achieving high data accessibility. If the novel replica allocation techniques can allocate replica without discussion with other nodes, as in a human friendship management, traffic overhead will decrease each node has a parameter d, the depth of SCF-tree. When Ni builds its own SCF-

tree, Ni first appends the nodes that are connected to Ni by one hop to Ni's child nodes. Then, Ni checks recursively the child nodes of the appended nodes, until the depth of the SCF-tree is equal to d.

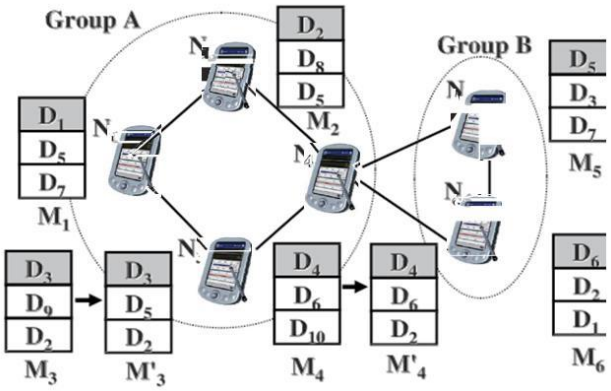

Fig 1: Example of selfish replica allocation

Fig.1 illustrates the network topology of some SCF-trees of N1 and N2 in Fig.1. In this example, we assume that all nodes are non selfish nodes for simplicity. As can be seen in Figs. 2b and 2c, the SCF-tree may have multiple routes for some nodes from the root node; N1 has two routes to N2 when N1 sets its own parameter d to be 4.

Since the multiple routes confer high stability. We allocate more replicas to the nodes that have multiple routes from the root node. At every relocation period, each node updates its own SCF-tree based on the network topology of that moment.

ALLOCATING REPLICA

After building the SCF-tree, a node allocates replica at every relocation period. Each node asks non selfish nodes within its SCF-tree to hold replica when it cannot hold replica in its local memory space. Since the SCF-tree based replica allocation is performed in a fully distributed manner, each node determines replica allocation individually without any communication with other nodes. Since every node has its own SCF-tree, it can perform replica allocation at its discretion. Since we assume that a node can use some portion of its memory space selfishly, we may divide memory space Mi for replica logically into two parts: selfish area Ms and public area Mp.

PERFORMANCE EVALUATIONS

Simulation Environment

The default number of selfish nodes is set to be 70 percent of the entire nodes in our simulation, based on the observation of a real application. We set 75 percent of selfish nodes to be type-3 and the remaining to be type-2. Type-3 nodes consist of three groups of equal size. Each group uses 25, 50, and 75 percent of its memory space for the selfish area.Type-2 nodes will not accept replica allocation requests from other nodes in the replica allocation phase, thus being expected to create significant selfishness alarm in query processing.

Type-3 nodes will accept or reject replica allocation requests according to their local status, thereby causing some selfishness alarms in subsequent query processing. We evaluate our strategy using the following four performance metrics:

1. OVERALL SELFISHNESS ALARM:

This is the ratio of the overall selfishness alarm of all nodes to all queries that should be served by the expected node in the entire system.

2. COMMUNICATION COST:

This is the total hop count of data transmission for selfish node detection and replica

allocation/relocation, and their involved information sharing.

3. AVERAGE QUERY DELAY:

This is the number of hops from a requester node to the nearest node with the requested data item. If the requested data item is in the local memory of a requester, the query delay is 0.

We only consider successful queries, i.e., it is the total delay of successful requests divided by the total number of successful requests.

SIMULATION RESULTS

EFFECTIVENESS OF DETECTION METHOD

We first compare the overall selfishness alarm of DCG with that of DCGþ to demonstrate the effectiveness of our detection method. We expect that the overall selfishness alarm will be reduced in query processing by detecting selfish nodes effectively with DCGþ, since many selfish nodes will be removed from the replica allocation phase and many reliable nodes will serve data requests from nodes. However, recall that the selfishness alarm may also occur due to network disconnections.

COMMUNICATION COST

We evaluate several replica allocation techniques in terms of communication cost. Our intuition was that our techniques outperform DCGþ, while being inferior to SAF. This intuition is confirmed by the results. DCGþ shows the worst performance in all cases, since group members need to communicate with each other in detecting selfish nodes and allocating/relocating replica. We report that, on average, about 70 percent of total communication cost in the DCGþ technique is caused by replica allocation/relocation, while about 30 percent is caused by selfish node detection. As expected, SAF shows the best performance, since no detection of selfish nodes or group communication is made. Although SAF and DCG techniques show better performance than DCGþ in communication cost, they are expected to show poor performance in data accessibility in presence of selfish nodes. Interestingly, our analysis reveals that our techniques, which detect selfish nodes, considerably outperform DCG, which does not perform the selfishness detection procedure. This verifies the efficacy of our fully distributed way of detecting selfish nodes and allocating replica, i.e., no group communication. There is no decisive difference among our techniques, except that the eSCF technique shows the worst behavior. The other techniques (SCF, SCF-DS, and SCF-CN) show very similar communication cost, since they are all based on the same SCF-tree structure.

AVERAGE QUERY DELAY

As expected, the SAF technique shows the best performance in terms of query delay, since most successful queries are served by local memory space. Our techniques show slightly better query delay than does the DCG technique. The DCGþ technique shows the worst performance. This can be explained as follows: the distance in hop counts among group members in the DCGþ technique is longer than that in the DCG technique. Since most successful queries are served by group members in these techniques.

EFFECT OF COMMUNICATION RANGE

Finally, we examine the effect of communication range. In all cases, our techniques outperform DCG and DCGþ, while SAF shows the best performance in terms of communication cost and average query delay. As the communication range increases, the communication cost of all techniques increases at first, but it gets smaller from a certain point except SAF. When the communication range is smaller than a certain point, the communication cost increases as the communication range gets larger, since the number of nodes connected to each other increases and thus the communication cost caused by replica relocation increases.

CONCLUSION

In contrast to the network viewpoint, we have addressed the problem of selfish nodes from the replica allocation perspective. We term this problem selfish replica allocation. Our work was motivated by the fact that a selfish replica allocation could lead to overall

poor data accessibility in a MANET. We have proposed selfish node detection method and novel replica allocation techniques to handle the selfish replica allocation appropriately. The proposed strategies are inspired by the real-world observations in economics in terms of credit risk and in human friendship management in terms of choosing one's friends completely at one's own discretion. We applied the notion of credit risk from economics to detect selfish nodes. Every node in a MANET calculates credit risk information on other connected nodes individually to measure the degree of selfishness. Extensive simulation shows that the proposed strategies outperform existing representative cooperative replica allocation techniques in terms of data accessibility, communication cost, and query delay. We are currently working on the impact of different mobility patterns. We plan to identify and handle false alarms in selfish replica allocation.

REFERENCES

[1]. E. Adar and B.A. Huberman, "Free Riding on Gnutella," First Monday, vol. 5, no. 10, pp. 1-22, 2000.

[2]. L. Ander egg and S. Eidenbenz, "Ad Hoc-VCG: A Truthful and Cost-Efficient Routing Protocol for Mobile Ad Hoc Networks with Selfish Agents," Proc. ACM MobiCom, pp. 245-259, 2003.

[3]. K.Balakrishnan, J. Deng, and P.K. Varshney, "TWOACK: Preventing Selfishness in Mobile Ad Hoc Networks," Proc. IEEE Wireless Comm. and Networking, pp. 2137-2142, 2005.

[4]. R.F. Baumeister and M.R. Leary, "The Need to Belong: Desire for Interpersonal Attachments as a Fundamental Human Motivation," Psychological Bull., vol. 117, no. 3, pp. 497-529, 1995.

[5]. J. Broch, D.A. Maltz, D.B. Johnson, Y.-C. Hu, and J. Jetcheva, "A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols" Proc. ACM MobiCom, pp. 85-97, 1998.

[6]. G. Cao, L. Yin, and C.R. Das, "Cooperative Cache-Based Data Access in Ad Hoc Networks," Computer, vol. 37, no. 2, pp. 32-39, Feb. 2004.

[7]. B.-G.Chun, K.Chaudhuri, H. Wee, M.Barreno, C.H. Papadimitriou, and J.Kubiatowicz, "Selfish Caching in Distributed Systems: A Game-Theoretic Analysis," Proc. ACM Symp. Principles of Distributed Computing, pp. 21-30, 2004.