A logical structure based fault tolerant approach to handle leader election in mobile ad hoc networks

Bharti Sharma a,*, Ravinder Singh Bhatia b, Awadhesh Kumar Singh b

a DIMT, Kurukshetra, India
b NIT, Kurukshetra, India

Received 11 January 2015; revised 4 March 2015; accepted 15 March 2015

Abstract We propose a lightweight layered architecture to support the computation of leader in mobile ad hoc networks. In distributed applications, the leader has to perform a number of synchronization activities among participating nodes and numerous applications; hence, it is a stressed node and consequently prone to failure. Thus, fast and fault tolerant leader election is a major concern and popular area of research in distributed computing networks, in general, and wireless ad hoc networks, in particular. In the present article, we have proposed a fault tolerant leader election approach. More importantly, the nodes elect the leader quickly on the basis of local information only. The illustration includes suitable examples. The correctness proof and performance evaluation has also been presented.

© 2015 Production and hosting by Elsevier B.V. on behalf of King Saud University. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

1. Introduction

1.1. Background

A mobile ad hoc network (MANET) is easily installable and thus, a very economical and popular computing environment; however, it is highly constrained and challenging too. A portable node possesses limited computing power processors, small stable storage, thin battery for energy backup, and a short communication range. They can communicate only via message passing over wireless links. Nodes that are not in the transmission range of each other can communicate via message relay. The nondeterministic mobility pattern incurs concurrent and arbitrary topological changes. The topological changes become more frequent because of the dynamism in wireless links and limited availability of bandwidth. A highly performance retarding offshoot of this phenomena is the variable message delay. Hence, the distributed algorithms developed for static domain are not directly implementable in MANETs.

Nowadays, there are many popular distributed applications that are executed in MANETs, e.g. agreement problem, inter-node communication, data exchange, service request, data aggregation, key distribution, group communication, and privilege grant. In the execution of any such applications, especially in MANET-like fault prone environment, the leader is a critical component that is responsible to coordinate such
activities. Thus, it is very useful to have a leader to make critical decisions in an easy way. Therefore, finding a leader (a.k.a. coordinator) is a popular research challenge in distributed systems community.

1.2. The problem statement

The leader election is a very simple form of symmetry breaking and is a classic synchronization problem of distributed computing. An algorithm for choosing a unique process to deliver a particular service is called an election algorithm and the elected process is called leader. For example, in a centralized server architecture based mutual exclusion algorithm, the ‘server’ is chosen from among the processes. It is necessary that all the processes agree on the choice. Afterward, if the process that delivers the service of the server fails or wishes to stop providing the service then another election process instance is executed to choose a substitute leader. We say that a process initiates the election if it takes an action that begins the execution of a particular instance of the election algorithm. An individual process does not initiate more than one election process instance at a time; however, in principle, several processes could initiate concurrent election process instances. At any point in time, a process \( p_i \) is either a participant – meaning that it is involved in some execution of the election algorithm – or a non-participant – meaning that it is not currently involved in any election. An important requirement for the choice of the elected process is to be unique, even if multiple processes call elections concurrently. For example, more than one process could notice independently that a leader process has failed, and they initiate elections. In short, computing a leader is to eventually elect a distinguished node from a given set of nodes.

1.3. Related work and motivation

A detailed discussion on leader election problem and protocols can be found in many popular books (Ghosh, 2010; Attiya and Welch, 2004; Garg, 2004; Raynal, 2013). In order to solve the problem, several distributed algorithms have been proposed in the literature that include a few classic protocols (Chang and Roberts, 1979; Hirschberg and Sinclair, 1980; Garcia-Molina, 1982) and an interesting array of recently published protocols (Boukerche and Abrougui, 2007; Derhab and Badache, 2008; Haddar et al., 2008; Dagdeviren and Erciyes, 2008; Ingram et al., 2009; Raychoudhury et al., 2014; Raz et al., 2004; Sharma and Singh, 2011; Shirmohammadi et al., 2009; Singh and Sharma, 2011; Vasisht et al., 2003, 2004; Malpani et al., 2000; Jain and Sharma, 2012; Subathra et al., 2012). In general, the protocols developed for the conventional distributed systems fail to work efficiently when implemented in cellular and ad hoc network based modern computing systems. For instance, the leader election protocol for MANETs needs to be message efficient because the nodes possess limited energy. The message overhead can be controlled by imposing a cost-effective logical structure on the physical network. However, most of the above listed protocols don’t exploit this idea. Furthermore, the energy of the leader node dissipates faster than ordinary nodes, due to the coordination overhead that the leader handles. It makes the leader node susceptible to crash. Secondly, the absence of a leader leads to reduced utilization of nodes’ resources, application discontinuity, performance degradation, or aborts. We intend to prevent these problems and thus our motivation is not to develop yet another leader election protocol, rather to design an election protocol that can choose the leader in a faster way. In this paper, we propose a logical structure based light weight protocol to support the leader election in MANETs. Also, our protocol quickly finds a replacement in the event of leader failure.

2. The protocol concept

2.1. The system settings

We make the following assumptions about the system architecture. In our illustration, the network is assumed to have a finite number of nodes distributed in a geographic region. A node has a universal identifier (UID) (called ID, for short) represented by a binary sequence that is unique and constant throughout the network lifetime. A node’s ID may be, for example, its MAC/IP address or CPU ID (Zeng et al., 2010). The unique ID assignment is a non-trivial problem and beyond the scope of this paper. Also, the nodes are assigned certain weights depending on the quality related attributes, like node’s residual power, computational capability, speed etc. The node IDs are used as a tie-breaker among equal weight nodes. The nodes may crash and recover autonomously. The communication links between nodes are assumed to be bidirectional FIFO and guarantee a message delivery if the corresponding sender and receiver remain connected during the message propagation. The message propagation delay is nondeterministic however bounded. Each node has a sufficiently large receive buffer; hence, it never suffers buffer overflow.

We propose a layered architecture, as shown in the following Fig. 1, to support leader election in MANETs.

- At the lowest layer, a clustering algorithm divides the MANET into balanced clusters, using a modified version of the previously designed merging clustering algorithm (MCA) (Dagdeviren et al., 2005) and it constructs a heap of cluster nodes.
- The second layer implements the ring formation (RF) algorithm which constructs the virtual ring of cluster heads. Further, the ring members are represented in the form of a heap.

![Figure 1 The layered architecture.](image-url)
دریافت فوری
متن کامل مقاله

امکان دانلود نسخه تمام متن مقالات انگلیسی
امکان دانلود نسخه ترجمه شده مقالات
پذیرش سفارش ترجمه تخصصی
امکان جستجو در آرشیو جامعی از صدها موضوع و هزاران مقاله
امکان دانلود رایگان ۲ صفحه اول هر مقاله
امکان پرداخت اینترنتی با کلیه کارت های عضو شتاب
دانلود فوری مقاله پس از پرداخت آنلاین
پشتیبانی کامل خرید با بهره مندی از سیستم هوشمند رهگیری سفارشات