An Efficient Leader Election in Peer-to-Peer Systems

A. A. Obeidat

Leader (coordinator) election is one of the most frequently encountered problems in Peer-to-Peer (P2P) applications. In collaborative applications, a leader mediates synchronization, consistency, sequencing and load balance. In different types of file systems, the leader may be used to eliminate redundant network traffic. The solution is to select a single peer that reports a located object matching a query. There have been a number of techniques which could be exploited by the leader election. However, these techniques tend to ignore the bandwidth bottleneck and congestion created by the concurrent peers in a session or the total network resource usage. Network congestion and resource usage efficiency are always among the most important issues in networking. These techniques assume all nodes vote (directly or indirectly) on the choice of each leader. These algorithms are not scalable because they require broadcasting or passing a token to all nodes. The best known election algorithms, electing a leader (typically the node with highest ID number) under various fault tolerant scenarios, such as Ring, Bully, etc[1].

The proposed method is based on characters of p2p system that all nodes have the same abilities and responsibilities. We propose a method to assign a leader to a set of nodes s_j having the same interest *X* in the environment of the P2P systems, for example, leader to the set of nodes containing replicas of some resource. Each set s_j has a unique identifier k_j. The identifier k_j of the set is computed by hashing the interest *X*. The hash is computed using a collision resistant hash function (e.g., SHA-1[2]). The node with an identifier v_j numerically closest to the k_j acts as a leader for the associated set s_j. In the algorithm every member of the set determines the leader by the overlay which always routes the messages to the node with identifier which is numerically closest to key of set. Therefore, the message complexity and overhead for proposed algorithm is O (1). When a given leader dies, other leader should quickly take over its functions. In algorithm, the failure of the leader is solved by routing in p2p system. After crashing the leader, the messages from members are routed by overlay to the node with identifier that is numerically closest to identifier of the set k_j. So the failure of the leader does not influence on stability of the work of the network. The leader is mobile between the active nodes in system. Therefore, the algorithm is able to handle the high rate of the churn in p2p systems.

Usually, the correctness of the election algorithm is proved by satisfying the following two conditions: a *safety* condition which implies that if there is one node whose status is leader then the status of the rest nodes in the system is non-leader and a *liveness* condition meaning that once the election starts, one node becomes coordinator finally. It is easy to see that the algorithm is correct, the first condition depends on the basic of algorithm that the leader is always the node from system with identifier that is numerically closest to identifier of the set k_j. The second condition is satisfied because the members of the set always ask the overlay to send the requests to the node which is more close to the identifier of the set. So, while there are nodes in the system, the leader exists.

Therefore, a method to assign a leader for a set of nodes having the same interest is presented with efficient performance; its complexity is O (1). In which the assigning of the leader performs autonomy (fully distributed) and the method may be applied in any of structured p2p systems.

References

1. George Coulours, Jean Dollimore, Tim Kindberg. Distributed systems. Addison-wesley,4th edition.2005.pp.471-479.
2. FIPS 180-1, “Secure hash standard,” Tech. Rep. Publication 180-1, Federal Information Processing Standard (FIPS), National Institute of Standards and Technology, US Department of Commerce, Washington D.C., April 1995.