

The Bio-Networking Architecture

Bi-weekly report #8 (September 16, 2002): Distributed Discovery

PI: Tatsuya Suda (suda@ics.uci.edu)

University of California, Irvine

<http://netresearch.ics.uci.edu/bionet/>

Introduction

In the Bio-Networking Architecture, a network application is implemented as a decentralized collection of autonomous cyber-entities called *cyber-entities*. One of the challenges with such a distributed framework is discovery of cyber-entities. Discovery provides the ability for applications to locate specific cyber-entities that may represent available services, information, or users.

In the bi-weekly report submitted on June 24, 2002, the PI described two new discovery mechanisms developed with support from DARPA. In this report, the PI focuses on evaluating the dynamics of one of the two discovery mechanisms described in the report submitted on June 24, 2002. In this discovery mechanism, each cyber-entities contains a set of keywords, and discovery involves locating cyber-entities that match some particular keywords. Cyber-entities also contain a limited set of relationships (links that include information about other cyber-entities) to other cyber-entities. These relationships between cyber-entities together form a network on which discovery queries are forwarded. The discovery mechanism in this report forwards discovery requests multiple hops and returns discovery results along the same path.

In this report, we focus on the behavior of the proposed discovery algorithm in dynamic environments. Dynamics in the environment we consider in this report includes cyber-entity availability, failures during discovery processing, changes in user demand, and changes in content provided by the cyber-entities. A cyber-entity's availability may change between an available or unavailable status due to unpredictable network-level connectivity or due to removal of the cyber-entity from the network. During discovery processing, an available cyber-entity may not complete discovery processing for various reasons such as excessive load. These dynamic properties impact how a discovery algorithm performs and may impact design choices for a discovery mechanism.

Proposed Discovery Algorithm

The discovery mechanism in this report addresses this dynamic environment through maintaining a history value for each relationship and through introducing timeouts at each hop to recover from failures.

Relationship history summarizes information about how a relationship partner has performed in past discoveries and is defined as the ratio of successful discovery queries on a relationship relative to all the discovery queries forwarded on that relationship. History provides a method for handling dynamic cyber-entity availability. A cyber-entity that attempts forwarding a discovery request to an unavailable cyber-entity realizes the unavailability, and then decreases the history value of the relationship to the unavailable cyber-entity. After a relationship is determined unavailable, this CE may then attempt forwarding the discovery request along relationships to other cyber-entities, and for cyber-entities that successfully return discovery results, history for those relationships relatively increases. Accordingly, history provides a method to select for relationships to the more reliable/available cyber-entities.

Timeouts provide a method for discovery to handle failures within the discovery process. During forwarding of a discovery request or returning of discovery results, cyber-entities may become unavailable or overburdened resulting in a loss of the discovery request or discovery results. Cyber-entities that may be waiting for discovery results from the request would receive no notification of the failure resulting in a timeout. The timeout allows the discovery process to fail-over to another relationship continuing the search process.

History also provides a method to handle dynamic user demand. User demand at a point in time may be biased towards certain keywords within the network. Based on this biased user demand, history enhances the performance of the network by adapting relationships to better support the biased user demand. Over time, user demand may change to a new user demand, and history would again adapt relationships according to the new user demand, causing the performance of the network to shift towards better supporting the new biased user demand. Accordingly, history provides a method to adapt the relationship network relative to dynamic user demand.

Simulation Results

Simulations have been performed evaluating the performance of the discovery algorithm in several dynamic scenarios. In this report, simulations for two of these dynamic scenarios are described. In the first dynamic scenario, there exist two copies of each object, one reliable and the other unreliable. Reliable cyber-entities are always available for discovery, whereas unreliable cyber-entities are frequently unavailable. In Figure 1, the performance of the algorithm is compared between a static (all reliable) and dynamic scenario to determine how the algorithm is impacted by the unreliable cyber-entities. Performance of the discovery mechanism is worse in the dynamic scenario compared to the static scenario; however, the difference between the static and dynamic scenarios appears relatively small. These simulation results suggest that the proposed discovery mechanism is robust to unreliable cyber-entities.

In the second scenario, random discovery failures occur within the network. These discovery failures occur during discovery forwarding with probability 0.001 at all cyber-entities. Figure 2 compares the performance of the algorithm in a static scenario (no failures) and in this second dynamic scenario. The search time distribution remains approximately the same for the dynamic scenario; however, the success rate decreases by 5 percent with failures. These simulation results suggest that the proposed discovery mechanism is robust to random failures in discovery processing.

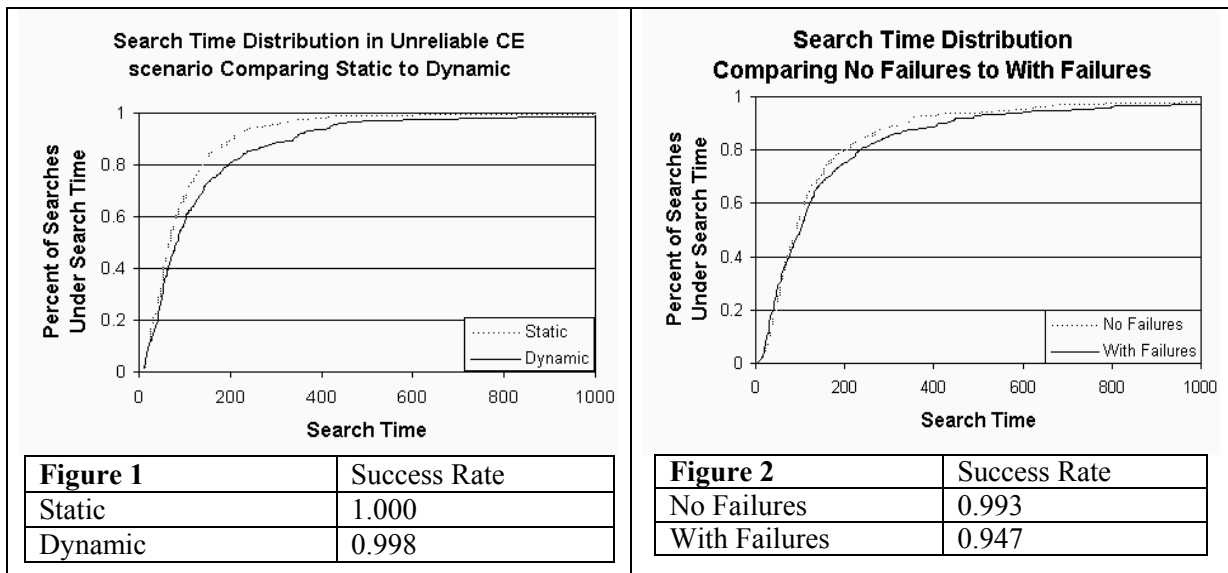


Figure 1	Success Rate
Static	1.000
Dynamic	0.998

Figure 2	Success Rate
No Failures	0.993
With Failures	0.947