Peer-to-Peer Systems

Seminar of Distributed Computing

DCG ETH Zürich

Nicole Hatt (hattn@student.ethz.ch)

December 2003

1 Summaries

1.1 Pastry

The paper [pastry] describes the discovery and the location of data and resources in a dynamic, decentralized and scalable network. Each node in the Pastry network is assigned an unique nodeId. Messages with a key are propagated to find resources in the network. Whenever a node is presented with a message and its key, Pastry automatically routes it to the node having the numerically closest nodeId to the key. To support the routing procedure, each pastry node maintains its routing state consisting of three tables providing information about nearby and numerically close nodes. Pastry has the ability to take into account network locality when routing messages: a message is not only sent to the numerically closest nodeId to the message key, it also chooses the physically shortest way to go there.

1.2 Viceroy

[viceroy] is a constant degree routing network of logarithmic diameter. It manages the distribution of data among a dynamically changing set of servers by employing a distributed hash table. Its main purpose is to efficiently look up resources without any central control. Each node in the system is randomly assigned two identifiers: an id and an integer identifying its level. The Viceroy network is composend of an approximate butterfly network, a ring connecting nodes in the order of their id's and level rings, where all nodes of the same level are connected in a ring. The butterfly network is a tree-like structure connecting a level ring with its upper and lower level. Therefore each node has a constant number of outgoing links. Routing consist of three phases, where the three kinds of links are followed to find the searched server.

2 Analysis

Both papers Viceroy as well as Pastry tackle the same problem of a peer-to-peer system: the storage and location of resources among a set of servers where nodes constantly join and leave the network. Identical to Chord, both provide a solution with a doubly connected ring as the underlying structure, where each node is randomly assigned an identifier using consistent hashing. The nodes are connected in a ring with their successor and predecessor nodes. However, Viceroy and Pastry differ in many points:

- The Viceroy approach improves on the existing ring structure. It combines an approximative butterfly network with the ring structure and achieves a constant number of outgoing links. For this reason, a join and leave operation requires a constant number of servers to change their states. Even though the outdegree of the node is fixed, the largest indegree might still be as large as $\log n$. In case of a node failure, $\log n$ links would have to be changed. Unfortunately [viceroy] only sketches an idea, which adds a background process to bound indegrees.

- Pastry routes the messages according to a scalar proximity metric. The message is forwarded to the numerically closest nodeId on the shortest way. In the neighborhood set each Pastry node keeps track of the physically closest nodes in the network. Also Viceroy aims to minimize the path length during the routing procedure by reducing the number of nodes involved. There exists no similar proximity mechanism as in Pastry to measure the physical distance.
- Viceroy is not able to deal with node failures. Only a follow-up paper goes further into that matter. Pastry handles node failures and leaves as soon as immediate neighbors state the failure/departure because they get no response when trying to contact the node. A special mechanism in Pastry allows the system to recover by replacing the failed node in the state tables of the immediate neighbors.
- Pastry has been implemented and simulated in a 100000 node network. As several experiments approve, Pastry behaves the way it is described. Applications are running on top of Pastry, as for example SCRIBE, a scalable publish/subscribe system, or PAST, a persistent storage utility. Quite contrary to Pastry, Viceroy is a theoretical approach of a distributed and scalable lookup service based on assumptions. There exists no implementation proving Viceroy's behaviour at the same degree as Pastry's implementation does.

The recent emergence of peer-to-peer applications like Napster, Gnutella and FreeNet certainly justifies the need for scalable lookup services managing the distribution of data. I personally have my doubts if Pastry or Viceroy really meets the needs of such applications in practice. Especially Viceroy, which is rather a theoretical and scientific approach than a commercial solution. It is obvious that a peer-to-peer routing network, which assumes no node to fail and no leave operation to overlap with a join operation has no chance to be brought to market. Even though Viceroy brings up very interesting aspects, such as the composition of the butterfly with the ring level links to achieve constant degree, many improvements are necessary to profit from it in practice. Because of the fact that Pastry provides mechanisms to deal with node failures and chooses the shortest distances during the routing procedure it fulfills more requirements than Viceroy. Experiments are evidence of Pastry's scalability and repair facilities. Nevertheless, the following issues can be criticised:

- The arrival of a new node in the Pastry network described in [pastry] is based on the assumption that any joining node initially knows about a physically close node already in the system. In my opinion this assumption is hard to fulfill in reality. Of course, a node could be found using IP multicast for example; but there is no guarantee that it always works. If a node knows about one nearby node, why should it not know about two or more physically close nodes when entering the network? This leads to the conclusion that the same mechanism could be used to initialize the neigborhood set.
- Despite the fact that Pastry is able to deal with node failures there is no protection against malicious agents in the system. What happens if a node pretends to be another node by manipulating its nodeId? What, if the messages are not forwarded correctly? To relativise this point, there is also to state that bringing security in a peer-to-peer system is a very demanding task and would lead far beyond the scope of the papers. One solution, for instance, is to add a central server for security reasons, which of course is the opposite of what a peer-to-peer system wants to achieve.

References

[pastry] Antony Rowstron, Peter Druschel: *Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems*; 18th IFIP/ACM International Conference on Distributed Systems Plattforms,2001

[viceroy] Dahlia Malki, Moni Naor, David Ratajczak: *Viceroy: A Scalable and Dynamic Emulation of the Butterfly*;