Performance Evaluation of a P2P Overlay Brokerage Network

Nikolaos Dellas, Sofia Kapellaki, Eleftherios Koutsoloukas, George N. Prezerakos[1], Iakovos S. Venieris
National Technical University of Athens, School of Electrical & Computer Engineering
9 Heroon Polytechniou str., 15773, Athens, Greece
{ndellas, sofiak, lefterisk, prezerak}@telecom.ntua.gr, venieris@cs.ntua.gr

Abstract **- Simplicity utilizes a self-managed peer-to-peer overlay broker network where good performance in terms of physical network utilization, event dissemination latency and routing state kept in brokers is desired. This paper is concerned with the performance evaluation of the proposed architecture that has been adopted within the Simplicity IST project. It focuses on the performance impact of the logical broker topology to physical network topology mapping. A custom discrete-time event-based brokerage level simulator has been developed and used for the performance studies. A key finding of our work is that the performance of publish/subscribe systems can be significantly improved if the mapping between the logical and the physical broker topology takes into account the publisher/subscriber relations.**

Key-Words **– P2P, publish/subscribe, overlay, brokerage networks, simulation, performance**

1 Introduction

Peer-to-peer (P2P) applications became widely known in 1999 with its first popular representative, the audio file sharing system Napster. Following Napster, a significant number of file sharing applications were developed and research interest in P2P concepts grew, not only for file sharing purposes but also for areas like distribution of computing power and storage, handling of complex processing jobs and instant messaging. The new generations of advanced P2P applications is currently attributed with a large portion of Internet traffic. To be more specific, almost 80% of the measured TCP traffic of a high speed IP backbone network connecting several ADSL areas can be attributed to P2P applications [2].

The term P2P refers to a class of systems and applications that employ distributed resources to perform a critical function, such as distributed computing, data/content sharing, communication and collaboration or any other service, in a decentralized manner [5]. The resources include computing power, data, bandwidth etc. The most important property of P2P systems is the absence of a "dedicated server" concept; participants in the P2P network are considered both clients and servers and there is a balance in the distribution of resources throughout the network.

Large-scale P2P systems make use of the publish/subscribe communication paradigm [3]. Providers of information advertise their presence to a network of interconnected nodes, where consumers of resources announce what kind of resources they need. It is impossible for every node in such systems to possess significant knowledge about the whole system topology; only the knowledge of a small number of neighbor nodes should be enough to enable the system to successfully match subscribers to appropriate publishers. Based on this set of neighbors, every node can route the published events to their destination. Thus an overlay network is formed, the resulting topology includes the graph of nodes, their interconnections and the neighbor relations between the nodes.

In the context of various P2P systems, nodes can be considered as communication brokers, since they receive and forward requests for resources in the system, on behalf of other nodes, participating this way in the discovery phase of resources for which they do not have immediate interest. In this context, the network of interconnected nodes can be seen as an overlay broker topology which guarantees the existence of a path between all the participating nodes, in the phase of announcement and discovery of resources. These broker overlay topologies are used for the delivery of published events to subscribed peers. The non efficient mapping between the overlay broker topology and the physical network topology is one of the most important factors of performance degradation in this kind of systems [4]; two neighbor brokers at the overlay topology may have many network layer hops between them. The impact of this mapping in the case of the SIMPLICITY brokerage architecture is studied in this paper.

The Simplicity brokerage architecture is briefly introduced in section 2 as an example of an overlay P2P system. In section 3 the related work in broker to network layer mapping and other crucial factors of P2P systems simulation is presented. In section 4 the simulator developed in the framework of the IST SIMPLICITY project [1] for the study of the brokerage topology is described. Some initial results obtained by the use of this custom simulator are presented in section 5. Finally, conclusions and future work are summarized in section 6.

[1] George N. Prezerakos is also with the Technological Education Institute (TEI) of Piraeus, Dpt. of Electronic Computing Systems, 250 Thivon Av. & Petrou Ralli, 122 44, Athens, Greece.

2 Simplicity Brokerage Architecture

IST SIMPLICITY project [1] attacks what we consider to be a significant hindrance in user's acceptance of a modern ICT (Information Communication Technology) environment, complexity. It is a distributed system employing state of the art context awareness and policy based adaptation concepts together with the novel idea of a "Simplicity Device", in order to create an easy to use, self configuring computing environment for users. The Simplicity Device can be seen as a ubiquitous ICT passport that the user always carries around with him, storing credentials for users' accounts and services, personal preferences and device, service and network profiles. The Simplicity Device, coming in different hardware realizations such as a JavaCard [17] or SIM cards into Bluetooth / Java enabled mobile phones, should be pluggable to any Simplicity enabled terminal. From there on, we foresee a network of devices and home appliances, as well as network side services and applications, each equipped with a Simplicity Broker, an intermediate entity between the specifics of a hardware and software environment and the Simplicity world. The network of Simplicity brokers around the user is what facilitates the delivery of modern networked context aware services by exhibiting, among others, an asynchronous event based communication scheme, resource discovery capabilities and direct broker to broker interaction in a P2P fashion.

For better understanding of the performed simulations a brief description of the overall Simplicity system architecture is essential [12].

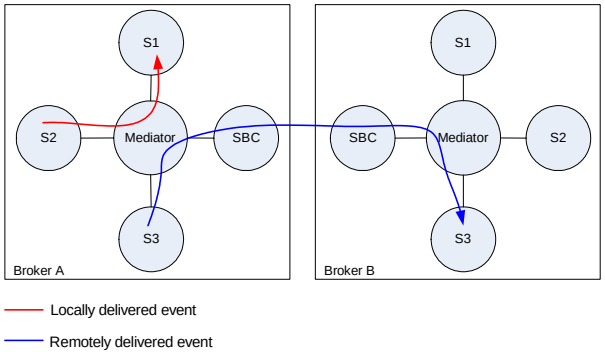

Fig. 1 Simplicity Architecture

In an abstract high level the Simplicity system can be thought of as a collection of event brokers. Each broker consists of a Mediator and one or more entities that produce and receive events (Fig. 1). These entities, called subsystems, offer different functionalities and can be seen as publishers and subscribers of events since they operate in a publish/subscribe manner. Some subsystems generate events and some others consume them based on subscriptions that subsystems express as they register towards the Mediator. An event generated by a subsystem can be received by subsystems residing in the same broker or by subsystems residing in a remote broker. For this reason a subsystem that all simplicity brokers need to have for the communication between them is the Simplicity Broker Communication (SBC) subsystem. If an event is not subscribed by any of the subsystems attached locally to the origin broker, then the event is forwarded to the SBC for the discovery of the appropriate recipient. For this purpose broker and subsystem discovery phases may be required. A service discovery procedure is also needed for the discovery of remote brokers [13].

According to the description above, the Simplicity system can be thought of as a P2P system in which Brokers play the role of peers. The decision of implementing the Simplicity system as a P2P system introduces many advantages like overcoming scalability and deployment problems and the achievement of better load balancing. Architectures that use one or more centralized functional entities suffer from bottleneck problems either at that entity or at the links towards that entity. Also, for the real deployment of the Simplicity system an event-based middleware is preferred not only for scalability improvement but also for deployment problems depreciation. Finally, a P2P system that uses adaptive load balancing techniques, as we will show, performs better in terms of links utilization, event delivery times and collision avoidance.

3 Related Work

The problem of overlay broker topology to physical network topology mapping is strongly related to the new research topic of load balancing of P2P systems, which is a very important issue regarding the real deployment of a P2P system. The term load balancing is used in P2P systems to describe the distribution of the system resources to improve overall performance. One method to achieve this is the mapping between broker and network levels. Two other categories of load balancing techniques are the distribution of queries and the distribution of replicas [6]. These load balancing techniques can not be applied to all categories of P2P systems due to limitations introduced by each system's architectural design. For example, some systems, like the file sharing system Freenet, create object replicas to peers other than the peers that have published a request for that object, a case that did not incur on the Simplicity system.

The mapping problem has been comprehensively studied in the case of the Pastry system, where the network proximity properties of an overlay broker P2P system were thoroughly analyzed [7]. The Hermes system, an event-based publish/subscribe middleware architecture, constitutes an extension of the previously mentioned work. For the comparison of Hermes with a standard publish/subscribe system a simulation model has been developed and used [8]. As we will describe later, our Simplicity Simulator (SSim) is similar as far as configuration logic is concerned with the one used in the case of Hermes, but some crucial extensions gives us the opportunity for more detailed studies of our system's performance behavior. Our main difference to this approach is that the above mentioned systems use some kind of structure for the delivery of published events, while the Simplicity system is considered fully

unstructured.

To focus on work related to unstructured P2P systems, like Simplicity, a location awareness method has been recently proposed [4]. Every peer sends periodically ping messages to its neighbor peers and the neighbor of its neighbor peers, in order to measure response times. Based on these measurements, the peers locally rearrange their neighbor links. Although the produced adaptive topology exhibits low search latencies, all published events have the same priority, while Simplicity system has to face some high priority events (policy decision events for example). Additionally to the simulation studies performed in the P2P research area, analytical models have been also presented for the prediction of the number of nodes reached during a search in a P2P system [9]. The results of this work have been compared with the ones provided by a simulation of the well-known P2P system Gnutella. Although this work provides some results, it does not take into account both the logical and physical topologies. As we will show in our experiments, the impact of the mapping between these two topologies can lead to different performance results.

In the context of all the previously mentioned research work and also the work presented herein, the creation of a representative topology of a P2P system over a physical Internet topology is necessary. For our work, we chose one of the most widely used tools in the area of representative Internet topologies generators, BRITE [10]. This tool includes all the previous work performed on Internet topology generators and provides a graphical environment for the easy creation of network topologies. BRITE produces topology output in various formats, like NS and Otter import files. Otter [11], a topology display tool, has been used for the visualization of the network topology used in our performance evaluation scenario.

4 The Simplicity Simulator (SSim)

As it was mentioned previously, the Simplicity system is mainly constructed by brokers, each of them containing a Mediator and one or more attached subsystems. Based on this very abstract architecture, a discrete-time event-based simulator has been developed in Java. The basic aim of this simulator, that we call Simplicity Simulator (SSim), is to study the performance of some important functionalities of the proposed architecture. The SSim design approach does not limit its applicability only to Simplicity system studies. The proposed simulator architecture, with the optimizations made, gives the required simulation scenario execution time saving level of abstraction for large-scale networks simulation, while provides a user friendly configuration and analysis user interface.

The subsystems generate events based on some predefined distributions. These events are inserted into the simulator priority queue and are then forwarded into the next agent. The agent is a crucial component of our simulator. Any process of events is performed into agent components. For the Simplicity environment three types of agents are used: mediator, SBC and subsystem. Each Simplicity broker consists of one SBC and one mediator agent component and any number of subsystems. An abstract UML class diagram of SSim is presented in Fig. 2 and a short description of the main simulation classes follows.

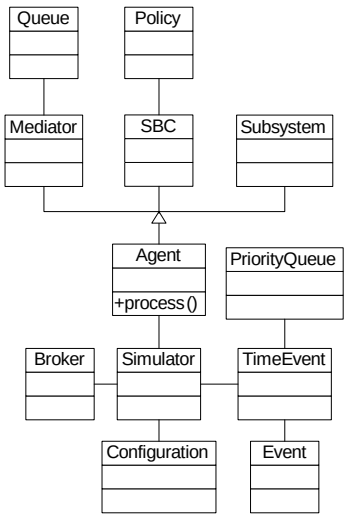

Fig. 2 Simplicity Simulation UML class diagram

The subsystems exchange Events based on simulator configuration parameters. The simulator's internal operation is based on TimeEvents that are exchanged between the Agents. TimeEvents are stored in a priority queue (class PriorityQueue) in a non-decreasing order according to simulation time. The processing of Events is performed by the process() method of the class Agent. This method is abstract within the class Agent and it is implemented within the derived classes. The Mediator class is responsible for the dispatching of events between local subsystems and simulates the event waiting time in the Mediator's event queue and the event dispatching latency. The type of mediator queue is specified by the Queue class. When an event is exchanged between two brokers the SBC subsystem is used. SBC is responsible for the simulation of the needed time for the delivery of the Event between the two brokers. For the specification of delivery policies, classes derived from class Policy can be used. Simulator class is the basic class of SSim. Classes derived from the Simulator class can be used for specific purpose simulations.

The configuration of simulation scenarios can be achieved manually through configuration files or by using the Simplicity configuration tool, called Simplicity Topology Generator (STG) (Fig. 3). With this tool various logical topology parameters can be specified. For example, as is shown in Fig. 3, the published and subscribed events of each broker can be configured along with the event generation times.

Before continuing to the presentation of some simulation results, some special SSim characteristics need more clarification. The construction of the physical and logical topologies is one of these characteristics. For the simulation of the Simplicity P2P overlay broker system two topologies are used, physical and logical topology. The former is a representation of an Internet topology, while the latter represents the logical interconnections

between peers. Previous studies have shown that these topologies follow small world and power low properties. Small world networks can be defined as networks with low number of edges[2] in the shortest path between two vertices, while the clustering coefficient is high. Other studies have also shown that the topologies produced by the BRITE topology generation tool following the transit-stub model have these properties. Transit-stub model mimics the structure of the Internet; first a number or independent networks (called autonomous systems, or AS – systems under an administrative authority) are created. Then these networks are interconnected to produce the generated topology.

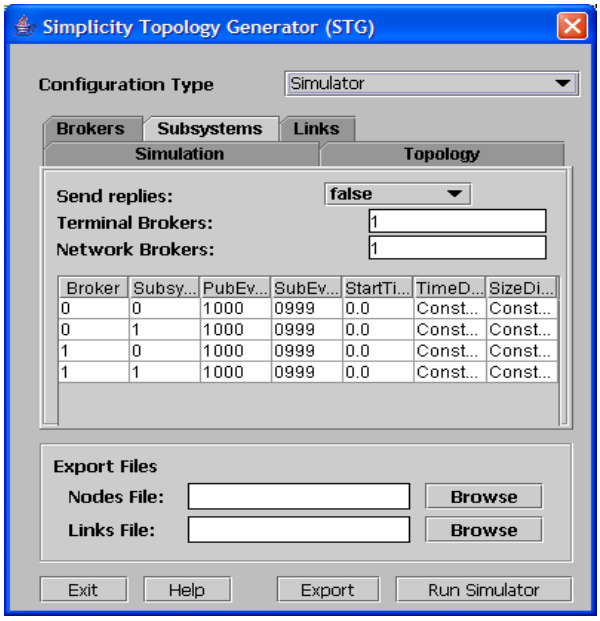

Fig. 3 Simplicity Topology Generator (STG)

For the production of the physical and logical topologies used in our studies, the BRITE topology generation tool has been used. Using the BRITE transit-stub model, two-level top-down topologies has been created, while the edges were produced by the Barabasi-Albert generator. For the physical network the link delays produced by BRITE were used. For the logical topology the communication delay between two neighbor edges was calculated (inside SSim) as the shortest path delay between the underlying physical nodes.

Another point that was given more attention is the scalability of the simulation. For this reason, the delays between the physical topology nodes are stored in an adjacent-list (also called edge-list) data structure. For every node, a list of the communication cost to its neighbor nodes is stored in this structure. To improve simulation performance, SSim uses internally two HashMap data structures; the first uses as a key the node id and as a value the second HashMap, while the latter uses as a key the other node id and as a key the communication cost. This data structure, although it is not optimized for communication cost queries, uses a small memory-footprint, a crucial factor in large-scale simulations.

Except the used data structure for the storage of the topology, the scalability of the SSim is addressed by hierarchical computation of the communication cost of event delivery. This communication cost is based on the inter-broker communication delays. The latter are computed by applying a shortest path algorithm into the underlying network topology. Due to the topology size this cost suffers from computation delays. So, some optimization of this process should be made. As it was mentioned before, the BRITE-generated network topology is a representative Internet topology, consisted of cluster of self-administrative nodes (called autonomous systems – ASs). The ASs are interconnected by a small number of links. For minimization of communication cost computation the routers used for this interconnection (which in the following description are called border routers) are identified and then the communication cost between them is computed before the execution of the scenario. These costs are stored in an array for retrieval time saving. So the communication between a pair of peers is computed as the minimal cost of the sum of costs from the source peer to a border router of the source peer's AS, the cost between border routers between the two ASs (if the peers belong to different ASs) and the cost between the target ASs border router and the target peer. The first and last communication cost computations are computed fast since the AS graph is composed of a few tens of vertices and this computation is internal to AS (the computation algorithm cost is dependent of the number of graph vertices).

The final point that received special care is the attachment of peers to the topology. The physical topology represents the Internet router level, while the nodes are attached to leaf-nodes (nodes that are not used for the routing in the AS backbone). The assumption taken is that leaf-nodes are the nodes that have small degree, or in other words, nodes with small number of directly connected nodes. For the simulation of peer arrivals and departures two special TimeEvents were used. For the maintenance of the connectivity of the overlay broker topology the set of peers produced by the BRITE topology generator is always present, while a set of dynamic peers arrive and depart from the system based on times specified by the specific scenario configuration.

5 Simulation Results

In this section some initial results taken using the SSim for the performance study of the overlay broker topology to physical network topology mapping are presented. Before the description of the actual used network topology, a short analysis of SSim characteristics for the simulation of P2P systems follows [14] [15].

The configuration parameters of the SSim scenarios under study are the following: number of brokers, number of subsystems acting as event publishers, number of subsystems acting as event subscribers, number of event

[2] Edges and vertices are terms often used in graph theory. In this paper these terms are used to refer to network (broker) layer links and nodes (peers) of physical (logical) topology, respectively.

types, number of different event types published by each publisher and number of different event types subscribed by each broker. For the following study only the values of the first three parameters varied; the other three parameters were set to the constant value of 1.

The metrics used in this study are the event notification latency and the hop count of event notifications, versus the number of subscribers and the policy used for the creation of the overlay broker topology. The policies used for this purpose were (i) random choice of neighbor brokers and (ii) closest broker in terms of network delay selection. For the latter policy, the performance improvement due to the knowledge of published event subscribers was examined. In addition, a policy, called optimal, was used as a reference to performance measurements; the notification latency of events is only limited by the underlay physical topology characteristics.

A representative Internet topology was necessary for the needs of our study. This topology should include not only the Autonomous Systems layer but also the node layer. Using the BRITE topology tool we have created a transit-stub network topology consisted by 10 AS each of which had 100 nodes. The topology produced by this tool has been adapted to the SSim configuration file format. This topology is depicted in Fig. 4 as it was produced by the network mapping tool Otter.

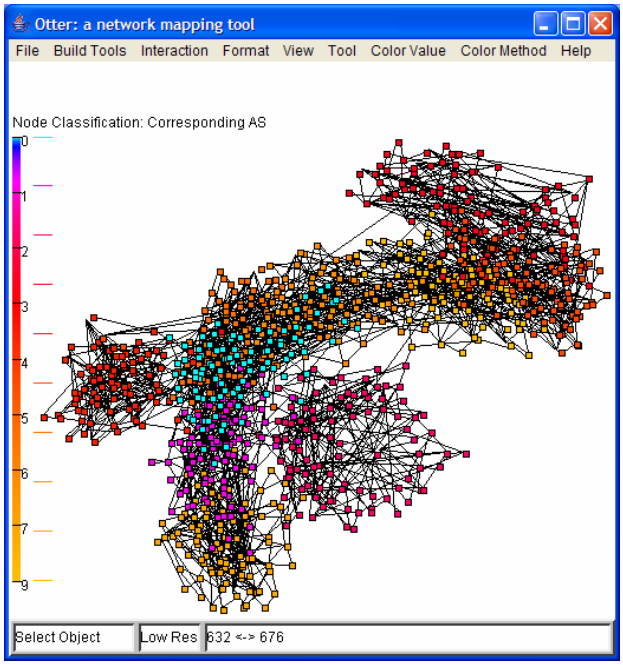

Fig. 4 A 10 AS and 1000 nodes network topology

5.1 Number of Edges vs. Event notification latency

The first performance study targets to the identification of the performance impact of the number of edges in the overlay network topology. To be more specific, the number of brokers was 50, the number of publishers 10, the number of subscribers 40 and a broker topology with 2, 4 and 40 edges has been created. The mean event notification latency has been compared with the optimal overlay broker topology. The collected results are summarized in Fig. 5.

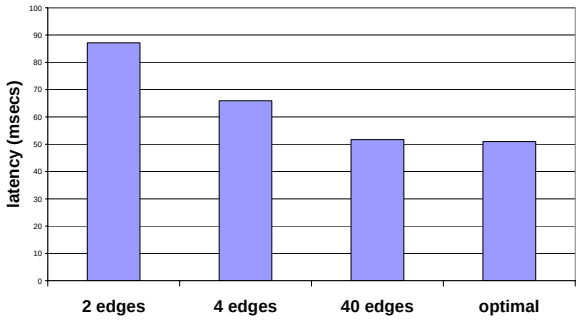

Fig. 5 Event delivery latency versus number of edges (random broker topology creation)

It is evident that the notification latency decreases as the number of edges increases. The disadvantages of this approach are the management overhead that it inflicts and the augmented size of the notification routing table. From Fig. 5 it is also shown that in the case of 2 and 4 edges the notification latency is 80% and 30% respectively greater than in the case of the optimal broker topology. An overlay topology with 40 edges performs like the optimal topology due to the used values of the parameters. A fully connected overlay broker topology was created and so the event delivery is determined only by the underlying physical network characteristics.

5.2 Number of subscribers vs. Event notification latency

The next study concerns the impact of the number of subscribers to the event notification latency. An overlay topology with 4 edges has been used and the number of subscribers was 10, 20, 30 and 40. The collected results are depicted in Fig. 6.

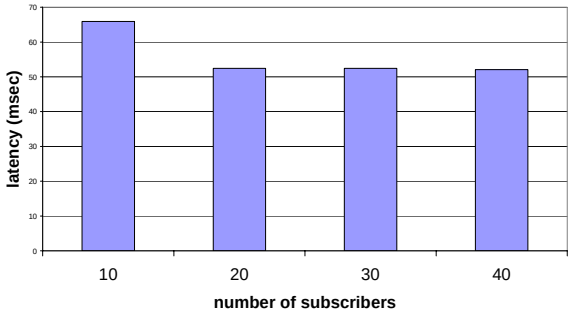

Fig. 6 Event delivery latency versus number of subscribers (random broker topology creation)

It is obvious that the more the subscribers are, the less the latency of event notifications delivery is. These results have been produced in the case of delivery to the closest broker. The results are similar also in the case of delivery to all subscribers of an event type.

5.3 Overlay Broker Topology Creation Policies

In this scenario, three policies have been used for the overlay broker topology creation: “random”, “closest broker” and “optimal”. Assuming that in all of the above policies the brokers are inserted sequentially and that the neighbor associations are made on the start-up phase of the simulation scenario execution, the “random” generation policy means that each broker chooses randomly its neighbor brokers. In the “closest broker” generation policy the set of neighbor brokers is the closest in terms of time needed for inter-broker communication. Finally, the “optimal policy”, which assumes that event delivery is limited only by the underlying network layer delays between any pair of brokers, is used as a reference for the following comparison. Initially the impact of the subscribed brokers number on event notification latency with different topology creation policies was examined. The collected results are summarized in Fig. 7.

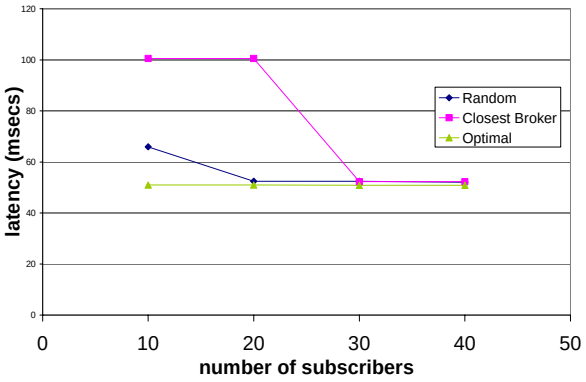

Fig. 7 Event delivery latency versus number of subscribed brokers for different broker topology creation policies

From the results it is obvious that the broker topology to physical topology mapping plays a critical role in determining the performance of the event delivery system. This performance degradation occurs mainly when the number of subscribed brokers is small compared to the total number of brokers. From a first look the results presented in Fig. 7 seem counter-intuitive; the “closest broker” policy suffers of greater delays than the random policy. A carefully look at the definitions of these policies gives as the answer; although the broker neighbor set has been optimized, the event forwarding process was not considered. When the number of event subscribers is a small portion of event brokers each event needs to be forwarded many times and so extra delay is inflicted. On the random policy case, these forwarding steps are less in number. Another observation is that when the number of subscribed brokers is greater than 50% of the broker population, all broker topology creation algorithms have the same performance. This result is self-explained; the number of brokers having event subscribes for this event type reaches the number of system brokers and so a response is returned by a broker close enough to the event publisher.

In the previous simulation scenario, the effect of the number of overlay topology hops needed for the delivery of the event on the mean event notification latency has not been examined. This effect is shown in Fig. 8, in the case of “random” and “closest broker” overlay topology creation policies.

With the random broker topology creation, event delivery latency strongly depends on the mean number of hops on the broker layer. In contrast, with the closest broker policy, brokers are always close to each other in terms of broker layer hops and so the latency doesn’t depend on the number of broker layer hops.

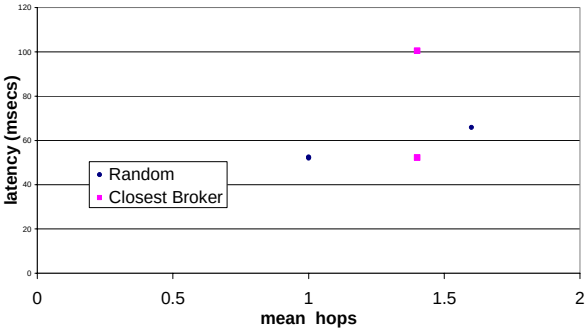

Fig. 8 Mean event delivery hops versus delivery latency

Fig. 8 depicts the impact of the overlay broker topology to physical network topology mapping. Two brokers that are close to each other at the broker topology level may not be so close in the physical network topology. In this simulation set, the network latency was measured against the number of neighbor brokers both for the “random” and “closest broker” policies. Two simulations were conducted for every policy, one with a large number of neighbor brokers and one with a smaller number of neighbor brokers. It is evident from the results obtained in Fig. 8 that in the random policy case, a larger number of neighbor brokers resulted in smaller mean number of event hops and that there was a small improvement of the time latency at the network level. The closest broker policy case however did not show any decrease of the mean event hops but there was a considerable improvement in time latency terms. This simulation set shows that improving the number of hops in the overlay broker topology alone is not sufficient to improve the time latency; the underlying network has to be considered as well. If the policy for the creation of the broker topology is not sophisticated enough, performance degradation might occur. This is shown in the case of the closest broker policy. Even when brokers choose the closest brokers (in terms of network latency) as neighbor brokers, system performance falls.

In the following the impact of event subscription knowledge on the performance of event deliveries was considered. In this experiment the closest broker policy (used in the first scenario) is compared with the closest brokers-subscribers only policy. The results of this study are presented in Fig. 9, along with the optimal broker topology scenario results.

From the following results it is obvious that considering the set of subscribers allows more sophisticated overlay broker topologies with almost optimal performance.

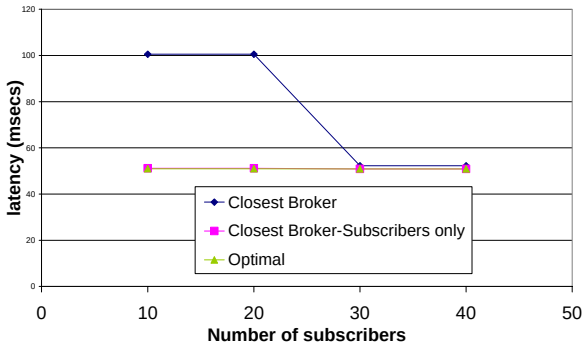

Fig. 9 Closest broker – subscriber only policy versus other broker topology creation policies

6 Conclusions

In this paper the performance impact of overlay broker topology to physical network topology mapping has been examined. After considering the Simplicity system in a P2P overlay broker manner, a custom simulator has been implemented and used for the collection of topology mapping performance related results.

The experiments described in this paper are at an initial stage but give an overview of the performance impact that some parameters have. These results also can lead to the creation of an adaptive overlay broker topology creation and maintenance algorithm, not only for the Simplicity like systems, but also for every P2P system. A good starting point is the algorithm proposed recently for efficient search in unstructured P2P systems [16], which aims at overcoming the disadvantages of flooding search methods, a serious reason for performance degradation of P2P systems. Even though in [16] the adaptation of overlay topology based on previous relevant query results has been examined, the underlying network topology characteristics haven't been taken into consideration. It was assumed that all links had equal costs, which was not a very realistic scenario.

Future work includes the usage of more topologies for the study of the overlay topology to network topology mapping. In addition more than one event types are going to be used for more realistic scenarios. These experiments can lead to a better understanding of the critical factors that affect this type of load balancing.

Acknowledgement

This work has been performed in the framework of the IST project IST-2004-507558 SIMPLICITY, which is partly funded by the European Union.

References

[1] IST project SIMPLICITY web page: http://www.ist-simplicity.org/

[2] N. Azzuna, F. Guillemin. "Experimental analysis of the impact of peer-to-peer applications on traffic in commercial IP networks", *European Transactions on Telecommunications*, Vol. 15, Issue 6, 2004, pp. 511-522.

[3] P. Eugster, et al. "The Many Faces of Publish/Subscribe" *ACM Computing Surveys*, Vol. 35, Issue 2, 2003, pp. 114-131.

[4] Y. Liy et al. "Location awareness in unstructured peer-to-peer systems", *IEEE Transactions of Parallel and Distributed Systems*", Vol.16, No.2, 2005, pp. 163-173.

[5] D. Milojicic, et al. "Peer-to-Peer Computing", HP Laboratories Palo Alto, *report HPL-2002-57*, March 8, 2002.

[6] Q. Lv, et al. "Search and replication in unstructured peer-to-peer networks", in Proc. of the *16th annual ACM International Conf. on Supercomputing (ICS '02)*, New York, USA, June 2002.

[7] M. Castro, et al. "Exploiting network proximity in peer-to-peer overlay networks", *Technical Report*, MS Research Cambridge, 2002

[8] P. Pietzuch, J. Beacon. "Peer-to-peer overlay broker networks in an event-based middleware", in Proc. of the *2nd international workshop on Distributed event-based middleware*, San Diego, California, 2003

[9] R. Schollmeier, G. Schollmeier. "An analytical model for the behavior of arbitrary peer-to-peer networks", *European Transactions on Telecommunications*, Vol. 15, Issue 6, 2004, pp. 523-534.

[10] BRITE topology generator web site: http://www.cs.bu.edu/brite/

[11] Otter topology display tool web site: http://www.caida.org/tools/visualization/otter/

[12] D2101 "Initial System Architecture Specification", http://server.ist-simplicity.org/deliverables.php

[13] D3101 "Design of the Network Broker System", http://server.ist-simplicity.org/deliverables.php

[14] D2301 "Evaluation of System Mechanisms", http://server.ist-simplicity.org/deliverables.php

[15] D2302 "Enhanced Performance Evaluation of System Mechanisms", http://server.ist-simplicity.org/deliverables.php

[16] V. Cholvi, et al. "Efficient search in unstructured peer-to-peer systems", *European Transactions on Telecommunications*, Vol. 15, Issue 6, 2004, pp. 523-534.

[17] Java Card technology web page: http://java.sun.com/products/javacard/