Project Number 001907

DELIS
Dynamically Evolving, Large-scale Information Systems

Integrated Project
Member of the FET Proactive Initiative Complex Systems

2nd DELIS Newsletter

November 2004
DELIS – Dynamically Evolving, Large-Scale Information Systems

Integrated Project within the FET Proactive Initiative Complex Systems

This second DELIS newsletter is devoted to presenting our project and some of our achievements to the Complex Systems Community. The newsletter starts with a short description of our objectives. The main part presents abstracts describing some of our insights and results achieved during the first year of the project. More information can be found on our Web site http://delis.upb.de.

The DELIS Project

Our Mission. Information Systems like the physical Internet, the World Wide Web, telephone networks, mobile ad-hoc networks, or peer-to-peer networks have reached a level that puts them beyond our ability to deploy them, manage them, and keep them functioning correctly through traditional techniques. Reasons for this are their sheer size with millions of users and interconnected devices and their dynamics; they evolve dynamically over time, i.e., components change or are removed or inserted permanently. For such systems, we have to abandon the goal of global optimality.

Within DELIS, we therefore concentrate on developing self-regulating and self-repairing mechanisms that, on the one hand, are decentralized, scalable, and adapt to changes in their environments. On the other hand, these decentralized mechanisms have to lead to a globally acceptable behaviour, avoiding undesirable or unstable situations.

We believe that the combination of insights from statistical physics, market mechanisms, and biological and social behaviour with advanced algorithmic research in Computer Science is the right combination of expertise necessary to develop methods, techniques, and tools to cope with such challenges imposed by large scale information systems, and to contribute to the world-wide effort in complex systems research towards understanding the principles necessary to manage such systems.

Our Objectives. The goal of DELIS is to develop methods, techniques and tools to cope with challenges imposed by the size and dynamics of today’s and future information systems, in an interdisciplinary effort of Computer Science, Physics, Biology, and Economy. The objectives we have identified and want to address are:

- To understand the structure, self organization and dynamics of large scale information systems,
- to provide methods, techniques and tools for controlling and optimizing such systems,
- to apply methods from market mechanism and game theory in order to understand and to organize the competition for resources like bandwidth, computing power, data etc.,
- to demonstrate the capabilities of our methods, techniques and tools in two application areas of high scientific and economic impact:
  - A management platform for telecommunication transport networks that integrates the wide range of current and future data formats and services, and
  - a decentralized, self-organizing Web search engine based on a peer-to-peer architecture, resulting in higher quality answers to search requests than current centralized search engines.
We have structured our project in the following six subprojects:

**Subproject 1: Monitoring, Visualizing, and Analyzing Large Dynamically Evolving Information Systems**

Subproject Leader: Stefano Leonardi; Universita di Roma “La Sapienza”, Italy

The goal of Subproject 1 is to develop concepts, theoretical foundations, algorithms, tools, prototypes and software platforms in order to help us observing, measuring, analysing, debugging, visualizing, and understanding the behaviour and evolution of complex Information Systems. Because of the size, dynamics, and unpredictability of the changes and because of the limited availability of research data, this is an important and challenging emerging research area.

**Subproject 2: Structure and Self-Organization of Evolving Networks**

Subproject Leader: Fabrizio Davide; Telecom Italia Learning Services S.P.A., Italy

Future telecommunication networks will be larger, more complex and more dynamic than current ones. This trend will be accelerated by the spread of wireless and peer to peer networking, where new communications channels are constantly being established and disconnected and where traditional hierarchical management approaches are no longer valid. As a result the next generation of telecommunication networks will need autonomous, adaptive and decentralized management systems and tools. In this setting, the goal of Subproject 2 is to develop a theoretical and empirical understanding of network topology and dynamics, and to use this as a guide for the development of new management algorithms and tools.

**Subproject 3: Large Scale Optimisation**

Subproject Leader: Burkhard Monien; Universität Paderborn, Germany

Accurate modeling of scientific problems often leads to formulations of what are now called large-scale optimization problems, which usually consists of optimizing a nonlinear objective function under a system of linear or nonlinear constraints, often involving thousands of continuous and/or discrete variables. A very recent and prominent domain of application is that of large-scale networks (communication, transportation, etc), with the Internet as the prime example. Challenging problems within this context include: understanding and controlling the interaction of TCP flows, controlling robust network congestions, dealing with network survivability issues in case of cascading failures, and developing an information theory for the wireless network computing paradigm.

**Subproject 4: Game Theoretic and Organizational Economics Inspired Approaches**

Subproject Leader: Paul Spirakis; Computer Technology Institute, Patras, Greece

Most of the existing and foreseen complex networks are built, operated and used by a multitude of diverse economic interests. A prime example is the Internet, perhaps the most complex computational artifact of our times. The (possibly) selfish nature of the participating entities calls for a deeper understanding of the network dynamics in order to efficiently achieve their co-operation, by possibly considering bounded rationality aspects. Challenging problems include: Understanding and controlling complex networks, when network actors are perhaps selfish; designing efficient algorithmic cooperation among many selfish agents; understanding and affecting the dynamics of self-organization in large complex networks.
Subproject 5: Biology-Inspired Techniques for “Organic IT”

Subproject Leader: Ozalp Babaoglu; Universita di Bologna, Italy

Recent research has frequently pointed out the analogies between biological processes or organisms and large-scale information systems. Indeed, the term “self-managing information systems” is not a bad definition of “life” itself. In much of our engineering work we try to build technological systems with lifelike properties. For example, their functionality should be robust against partial damage to the system; we also want the system to be adaptive in the face of environmental challenges to its functioning, self-organizing (able to build itself and grow), and self-healing (able to repair large-scale damage). Finally, it is desirable that systems should be scalable or they should continue to function well, even when deployed on a worldwide scale. Society as a whole benefits enormously from systems that manage themselves. Hence it is important to develop the art of building self-managing information systems to the point at which it becomes engineering.

Subproject 6: Data Management, Search, and Mining on Internet-scale Dynamically Evolving Peer-to-Peer Networks

Subproject Leader: Gerhard Weikum; Max-Planck-Institut für Informatik, Saarbrücken, Germany

With the exploding number of information sources in the Internet and the high dynamics of their data, it will become absolutely crucial to factor large-scale community behaviour into a novel kind of system architecture in order to make this wealth of information effectively searchable, learn statistical and logical. Combinatorial models will have to be made, hypotheses will have to be evaluated, and ultimately knowledge will have to be extracted. Internet-based peer-to-peer systems are the most prominent example of dynamically evolving, large-scale complex systems. Current applications range from Grid computing to Napster-style large-scale file sharing. This subproject aims at more advanced information-centric applications, more specifically, data mining on evolving data and collaborative Web information search. The goal of this subproject is twofold: on one hand, to provide the basis for such an architecture, taking Internet-based peer-to-peer systems as the paradigmatic system to study, and on the other hand, to bring techniques from data mining and distributed data structures closer to what is necessary to analyze and manage large-scale systems that evolve quickly in time.

Project coordinator

Prof. Dr. math. Friedhelm Meyer auf der Heide
Heinz Nixdorf Institute, University of Paderborn
Fürstenallee 11
33102 Paderborn, Germany

Contact:
Email: fmadh@uni-paderborn.de
Telephone: + 49 - 5251 - 606480
Fax: + 49 - 5251 - 606482
http://delis.upb.de
Contents

Contributions of SP 1

BGPlay: Inter-domain Routing Visualization .............................................................. 3
Lorenzo Colitti, Giuseppe Di Battista, Federico Mariani, Maurizio Patrignani, and Maurizio Pizzonia

Traceroute-like discovery and exploration of networks ............................................. 7
Luca Dall’Asta, Ignacio Alvarez-Hamelin, Alain Barrat, Zuzana Beerliova, Felix Eberhard, Thomas Erlebach, Alexei Vázquez, and Alessandro Vespignani

Methodology for Estimating Network Distances of Gnutella Neighbors .................. 11
Vinay Aggarwal, Stefan Bender, Anja Feldmann, and Arne Wichmann

Contributions of SP 2

Platforms for Simulations of Complex and Mobile Ad Hoc Networks ....................... 15
Giannis Grammatikopoulos, Stefan Rührup, Christian Schindelhauer, and Klaus Volbert

A new Infrastructure for the Management of Networks and Grids based on Peer-to-Peer . 19
Giovanni Cortese, Felix Heine, Federico Morabito, and Fabrizio Davide

Contributions of SP 3

Typical Properties of Winners and Losers in Discrete Optimization ......................... 21
Rene Beier and Berthold Vöcking

Efficient Smart Dust Protocols ................................................................................. 23
Sotiris Nikoletseas and Paul Spirakis

A new approach to compute schedules for unrelated parallel machines .................... 27
Martin Gairing, Thomas Lücking, Burkhard Monien, Marios Mavronicolas, and Andreas Woclaw

Contributions of SP 4

A Group-Strategyproof Mechanism for Steiner Forests ............................................. 29
Jochen Könemann, Stefano Leonardi, and Guido Schäfer

Selfish Unsplittable Flows ......................................................................................... 33
Dimitris Fotakis, Spyros Kontogiannis, and Paul Spirakis
On the Evolution of Selfish Routing ................................. 37
Simon Fischer and Berthold Vöcking

Nash Equilibria in Discrete Routing Games with Convex Latency Functions ................................. 39
Martin Gairing, Thomas Lücking, Marios Mavronicolas, Burkhard Monien, and Manuel Rode

Contributions of SP 5

Producing Co-operative Peer-to-Peer Networks ................................. 41
Ozalp Babaoglu and David Hales

Structure in Software Graphs ................................................. 43
Ricard V. Solé and Sergi Valverde

Contributions of SP 6

Efficient Top-k Queries in Peer-to-Peer Information Systems ................................. 45
Gerhard Weikum, Sebastian Michel, Ralf Schenkel, Martin Theobald, and Peter Triantafillou

Harnessing Heterogeneity in Internet-Scale Peer-to-Peer Networked Information Systems ...................... 49
Peter Triantafillou and Nikos Ntarmos

A topographic view of epidemic spreading on networks ........................................ 53
Geoffrey Canright and Kenth Engø-Monsen
The Internet is administratively partitioned into networks, called Autonomous Systems (AS), or domains, where each AS is under a single administrative authority. Usually, an Internet Service Provider (ISP) controls one or more ASes.

Roughly speaking, at the AS level the Internet routing is described by a collection of sequences of ASes, called AS-paths. An AS-path $AS_1, AS_2, ..., AS_n$ says that the packets directed to $AS_n$ and originated by some device in $AS_1$ should traverse $AS_2$, $AS_3$, etc. in the specified order. The collection of all the AS-paths in use in the Internet at a given moment can be “merged” into a graph, called a routing graph.

To understand our goals, consider the Network Operating Center (NOC) of some ISP and suppose that the NOC wants to know which paths packets sent to $AS_x$, which is operated by the NOC, currently follow.

The BGPlay system displays the portion of the routing graph that describes how the traffic flows to $AS_x$ from a selected set of ASes which are considered, in some sense, “representative” of the entire Internet. It adopts specifically tailored techniques and algorithms to display the state of routing at specific points in time and to animate its changes.

BGPlay obtains routing data from well-known and publicly available sources, namely the Routing Information Service (RIPE NCC) [6] and the Route Views project (University of Oregon) [7]. Such institutions collect messages exchanged by inter-domain routers by means of the Border Gateway Protocol [5]. Their historical archives are used for network debugging purposes or scientific investigation and are updated in real-time.

The BGPlay system is currently hosted by RIS and Route Views as one of the tools for accessing the routing data they publish [1, 2].

Architecture and User Interaction

BGPlay has a three tier architecture which permits its deployment over the web and easy access to serveral data sources (see Figure 1). The middle tier is in charge of obtaining routing events from data sources, performing some computation on them and send them to the client which is in charge of performing visualization and user interaction.

To query BGPlay, the user connects to a web page which hosts BGPlay (for example, the instance deployed at the RIS [1]) and starts the BGPlay applet. The BGPlay query window (Fig. 2) appears, allowing the user to specify the prefix to examine, the time interval, and the observation points to use in the query (using the RRC00, RRC01, ... checkboxes).

When the user submits the query, BGPlay processes the request and displays the Animation window (Fig. 3), which presents the routing information. The left part of the window contains the time panel, which plots the number of events over time on a logarithmic scale. The bottom of the panel corresponds to the start of the query interval and the top of the graph to the end; the small blue triangle indicates the current time (initially, the start of the query interval). The user may jump to a specific instant within the query interval by clicking on the time panel.

The main part of the window contains the routing graph. Each number represents an AS, and the AS originating the prefix (in this case, AS 3333), is placed in the center of the graph and highlighted by a red
Figure 1: The BGPlay architecture

circle. The user may obtain the name and description of an AS by clicking on it. If desired, the position of any AS in the graph may be changed by dragging it with the mouse.

Each solid or dashed line represents a segment of an AS path seen by the data sources. A path starts in the originating AS and stops in the AS of a collector-peer; there is one path for every such peer which has an entry for the queried prefix in its routing table. The paths which did not change during the query interval are drawn dashed, while the paths which did change are drawn solid, but the colour of the path itself has no special meaning: different colours are used only to ensure that each AS-path from a peer to the source AS can be unambiguously identified.

Note that the graph may contain isolated nodes which have no paths to the origin AS. This does not necessarily imply that these ASes do not have a path to the queried prefix: more usually these ASes do not contain collector-peers (and thus no information about their routing is known) and appear in the graph because they have been or will be part of a path which was in use in another moment of the query interval.

The bottom of the window contains a control panel which allows the user to toggle the display of route reannouncements, start a new query, and move through the sequence of events that occurred in the specified time interval. Both forward and backward movement is possible. As each routing event is displayed, BGPlay updates the routing graph with a smooth animation and displays information on the event itself in the upper part of the window. This includes the event identifier, a timestamp, the type of the event, the collector which recorded the event and the peer from which it was received, and, depending on the type of event, additional information.
Figure 2: The query window

Figure 3: The animation window
References

[1] BGPlay service hosted by the RIS project.  
http://www.ris.ripe.net/bgplay/

[2] BGPlay service hosted by the RouteViews project.  
http://bgplay.routeviews.net/bgplay/


http://www.ris.ripe.net/

http://www.routeviews.org/
Traceroute-like discovery and exploration of networks

Luca Dall’Asta¹, Ignacio Alvarez-Hamelin¹, Alain Barrat¹, Zuzana Beerliova², Felix Eberhard², Thomas Erlebach³, Alexei Vázquez⁴, and Alessandro Vespignani¹,⁵

¹ Université de Paris-Sud, France
Ignacio.Alvarez-Hamelin@lri.fr

² ETH Zürich, Switzerland
bzuzana@inf.ethz.ch efelix@student.ethz.ch

³ University of Leicester, UK
t.erlebach@mcs.le.ac.uk

⁴ University of Notre Dame, USA
avazque1@nd.edu

⁵ Indiana University, Bloomington, USA
alexv@indiana.edu

A significant research and technical challenge in the study of large information networks is related to the lack of highly accurate maps providing information on their basic topology. This is mainly due to the dynamical nature of their structure and to the lack of any centralized control resulting in a self-organized growth and evolution of these systems. A prototypical example of this situation is faced in the case of the physical Internet. The topology of the Internet can be investigated at different granularity levels such as the router and Autonomous System (AS) level, with the final aim of obtaining an abstract representation where the set of routers (ASs) and their physical connections (peering relations) are the vertices and edges of a graph, respectively. As accurate maps of the Internet are not available, researchers follow the approach of acquiring local views of the network from several vantage points and merging these views in order to produce presumably good approximations of the global map. Local views are obtained by evaluating a certain number of paths to different destinations by using specific tools such as traceroute or by the analysis of BGP tables. At first approximation these processes amount to the collection of shortest paths from a source node to a set of target nodes. The merging of several of these views provides the map of the Internet from which statistical properties of the network are evaluated.

Using this approach, a number of research groups have generated maps of the Internet, e.g. [8], that have been used for the statistical characterization of network properties. It has been observed that the Internet is a relatively sparse network with small diameter and heavy-tailed (power-law) degree distribution [9, 4, 10, 5]. The latter observation has generated significant activity in the field of network modeling and characterization [6, 2, 11].

The maps used in these investigations, however, are undoubtedly incomplete. Typical mapping projects are run from relatively small sets of sources whose combined views miss a considerable number of edges and vertices [5]. In particular, the local views often miss the lateral connectivity of targets and sample more frequently nodes and links which are closer to each source, introducing spurious effects that might seriously compromise the statistical accuracy of the sampled graph. These sampling biases have been explored in numerical experiments of synthetic graphs generated by different algorithms, e.g., in [7]. Interestingly, it has been shown that apparent degree distributions with heavy-tails may be observed even from homogeneous topologies such as the classic Erdős-Rényi graph model.

These problems have prompted us to conduct a thorough examination of how sampling biases affect the measurements of the Internet topology [1], and to define a concise theoretical framework for studying fundamental issues of network topology discovery based on merging of local views, leading to the design.
of optimized mapping algorithms [3]. Some results of these activities in Subproject 1 of DELIS are summarized in the following.

**Statistical approach to traceroute-based discovery**

We have performed a mean-field statistical analysis and extensive numerical experiments of shortest path routed sampling, used as a model for traceroute-sampling in different network models. We find an approximate expression for the probability of edges and vertices to be detected that exploits the dependence upon the number of sources, targets and the topological properties of the networks. The expression shows the dependency of the efficiency of the mapping process upon the number of sources, targets and the topological properties of the network. Moreover, the analytical study provides a general understanding of which kind of topologies yields the most accurate sampling. In particular, we show that the map accuracy depends on the underlying network betweenness centrality distribution; the heavier the tail the higher the statistical accuracy of the sampled graph.

We have substantiated our analytical finding with a thorough exploration of maps obtained varying the number of source-target pairs on networks models, with different topological properties. According to the theoretical analysis, both the total number of probes deployed and the topological properties seem to play a primary role in the understanding of the level of the efficiency reached by the mapping process. As a measure of the efficiency of the mapping in different network topologies, we have studied the fractions of discovered nodes and edges as a function of the degree, stressing the agreement with the theoretical predictions. Other interesting quantities, such as redundancy, transit frequency and traffic entropy, have been introduced in the study of the discovery process, with the aim of providing a complete framework for the study of sampling redundancy. The study of the connectivity distributions of sampled graphs shows that single source mapping processes face serious limitations in that also the targeting of the whole network results in a very partial discovery of its connectivity. On the contrary, the use of multiple sources promptly leads to obtained maps fairly consistent with the original sample, where the statistical degree distributions are qualitatively discriminated also at low values of target density. A detailed analysis of the behavior of the degree distribution as a function of target and sources has been carried out for sampled graphs with different topologies and compared with the insight obtained by analytical means.

We have also inspected quantitatively the portion of the network that is discovered in different mapping processes imposing the same density of probes to the network. We have found the presence of a region of low efficiency (fewer nodes and edges discovered) depending on the relative proportion of sources and targets. This low efficiency region however corresponds to the optimal estimation of the network average degree and clustering. This finding calls for a “trade-off” between the accuracy in the observation of different quantities and hints to possible optimization procedures in the traceroute-driven mapping of large networks.

**Summary and perspective**

The rationalization of the exploration biases at the statistical level provides a general interpretative framework for the results obtained from the numerical experiments on graph models. The sampled graph clearly distinguishes the two situations defined by homogeneous and heavy-tailed topologies, respectively. This is due to the exploration process that statistically focuses on high betweenness nodes, thus providing a very accurate sampling of the distribution tail. In graphs with heavy-tails, such as scale-free networks, the main topological features are therefore easily discriminated since the relevant statistical information is encapsulated in the degree distribution tail which is fairly well captured. Quite surprisingly, the sampling of homogeneous graphs appears more cumbersome than that of heavy-tailed graphs. Dramatic effects such as the existence of apparent power-laws, however, are found only in very peculiar cases. In general, exploration strategies provide sampled distributions with enough signatures to distinguish at the statistical level between graphs with different topologies.

This evidence might be relevant in the discussion of real data from Internet mapping projects. Indeed, data available so far indicate the presence of heavy-tailed degree distribution both at the router and AS
level. In the light of the results of the artificial sampling of random graphs, it is very unlikely that this feature is just an artifact of the mapping strategies. The major part of mapping projects are multi-source, a feature that we have shown to readily wash out the presence of spurious power-law behavior. The heavy-tail behavior observed in real mapping experiments is likely to be a genuine feature of the Internet.

On the other hand, it is important to stress that while at the qualitative level the sampled graphs allow a good discrimination of the statistical properties, at the quantitative level they might exhibit considerable deviations from the true values such as size, average degree, exponent of the degree distribution and clustering properties. In this respect, it is of major importance to define strategies that optimize the estimate of the various parameters and quantities of the underlying graph. In this work we have shown that the proportion of sources and targets may have an impact on the accuracy of the measurements even if the number of total probes imposed to the system is the same. For instance, the deployment of a highly distributed infrastructure of sources probing a limited number of targets may turn out as efficient as few very powerful sources probing a large fraction of the addressable space. The optimization of large network sampling is therefore an open problem that calls for further work aimed at a more quantitative assessment of the mapping strategies both on the analytic and numerical side.

Theoretical framework for network discovery

In a second line of research complementary to the statistical approach summarized above, we have defined a simplified theoretical framework that allows to study fundamental aspects of network discovery problems in a combinatorial model. While the statistical approach assumes that the measurements are performed at nodes that are selected randomly, here we consider incremental strategies that determine the location of the next measurement depending on the information about the network that has been discovered by the measurements so far. Our goal is to devise algorithms (measurement strategies) that discover the maximum amount of information using as few measurements as possible. We measure the quality of an algorithm using competitive analysis, a standard approach in the study of computational problems under incomplete information. This means that we compare measurement algorithms with the best possible strategy for any given network. We assume that the algorithm initially knows the set of nodes of the network, but has no information about presence or absence of links. A measurement at a node reveals all edges on shortest paths between that node and any other node; this is a simplified model of the information revealed by extensive traceroute-based experiments carried out from that node to all possible destinations. We consider the discovery of links and also the absence of links as main goals of the measurement process. Thus, a network is fully discovered only if all links have been discovered and all absent links have been confirmed as absent.

We have proved that determining the best possible (off-line) algorithm for a given network is NP-hard and even NP-hard to approximate within a sublogarithmic factor. For the on-line problem of discovering a network using the fewest possible measurements, any reasonable algorithm guarantees a competitive ratio of $O(n)$, where $n$ is the number of nodes of the network. We have been able to design a randomized algorithm with competitive ratio $O(\sqrt{n \log n})$, thus providing a substantial improvement. In addition, we have designed several strategies that perform very well in simulation experiments with a variety of network topologies. These strategies often require a surprisingly small number of measurements to discover large-scale networks; they appear to be very effective for power-law topologies such as the real Internet. In addition, we have related the optimal number of measurements needed to discover a network to various network quantities such as diameter and maximum degree. For certain topologies, such as trees, we have been able to characterize the optimal number of measurements precisely.

Summary and perspective

Our research into a theoretical framework of network discovery processes has shown that this problem can be studied rigorously with competitive analysis and that, in computational experiments, the designed measurement strategies perform significantly better than a random selection of measurement locations. This indicates that in practice, one should find the right balance
between statistical measurements carried out at random locations (which have the advantage that the measurement points can be selected independently and the measurements carried out in parallel) and optimized strategies that select the next measurement location based on already discovered information (which introduces a sequential dependence into the mapping process, but can potentially reduce the number of required measurements substantially). We believe that a good compromise between the two paradigms can lead to significantly improved network discovery methods.

Concerning the theoretical framework, future work will concentrate on other, more realistic models of measurements (instead of revealing all edges on shortest paths between the measurement node and the remaining network, a measurement could reveal only an arbitrary shortest-path tree, for example) and on a study of different measurement goals (instead of discovering all links, one could study the problem of discovering only the degree distribution or the diameter, for example).

References


[8] The National Laboratory for Applied Network Research (NLANR), sponsored by the National Science Foundation. (see http://moat.nlanr.net/).


Methodology for Estimating Network Distances of Gnutella Neighbors

Vinay Aggarwal¹, Stefan Bender², Anja Feldmann¹, and Arne Wichmann¹

¹ Technische Universität München, Germany
{vinay,anja,aw}@net.in.tum.de

² Max Planck Institut für Informatik, Saarbrücken, Germany
sbender@mpi-sb.mpg.de

In this paper we ask the question how much does the neighborhood selection process of a P2P protocol such as Gnutella respect the underlying Internet topology.

Introduction

In peer-to-peer file sharing systems, the neighborhood selection process is governed by system-specific metrics or arbitrary choices. Here, we ask the question how close is a P2P topology to the Internet topology, using Gnutella as our example overlay network. The answer can help us estimate the (in-)efficiency of using overlays.

Gnutella [1] is based on agents, called servents. According to its original design each servent maintains connectivity with a set of others by sending Ping messages, which are answered using Pong messages. Search queries are flooded within the Gnutella-network using Query messages and answered by Query Hits. To limit flooding Gnutella uses TTLs and message IDs. Each answer message (Query Hit/Pong) takes the reverse path of the corresponding triggering message. Due to scalability problems, later versions of Gnutella protocol [2] introduced a hierarchy which elevates some servents to ultrapeers, while others become leaf nodes. Each leaf node connects to a small number of ultrapeers while each ultrapeer maintains a large number of neighbors, both ultrapeers and leafs. To further improve performance and to discourage abuse, the Ping/Pong protocol underwent semantic changes. Answers to Ping messages may be cached and too frequent Pings or repeated Queries may cause termination of the connection or may be ignored by the receiver.

These changes have vastly improved the scalability of the Gnutella network [2]. Yet at the same time they pose a huge impediment to investigating the structure of Gnutella, which is based on the original semantic of the Ping/Pong protocol. Accordingly in this paper we investigate how to overcome these limitations to find neighbors in the Gnutella network and then explore their network distance in comparison to random node distances.

Identifying edges in a P2P network

In order to study how close the P2P topology is to the Internet topology we first need to identify a representative set of edges in the P2P network. Then we need to find a comparable set of edges in the Internet and a metric suitable for comparison.

The most obvious way of finding edges in a P2P network is to create some by participating. Yet these are not representative. They are highly biased by the location and the software of the participant. Rather we want to identify edges in the P2P network where neither of the two nodes is controlled by us. We refer to any two nodes connected by an edge as neighbor servents and those not involving a node controlled by us as remote neighbor servents.

Due to the changed semantics of the Ping/Pong protocol the simple crawling approach outlined in the last section is no longer sufficient. As Pong messages are cached and due to the rapid fluctuations in Gnutella
networks one cannot assume that answers to Pings with TTL equal to two (so called crawler pings) contain still active servents. They should, however, have been remote neighbor servents at some point. Note that leaf nodes are no longer reported in Pongs.

To cope with these complications we deploy a combination of active and passive techniques to explore the Gnutella network. From the passive technique (an ultrapeer servent) we gain a list of active servents. Using Queries with TTL value of two allows us to get a set of remote neighbor servents. Using Pings with TTL value of two results in a set of candidate servents. These are then contacted actively to further advance the network exploration. This approach allows us to discover edges in the Gnutella network that existed at some point but it does not guarantee that they still exist. In the future we plan to enhance our strategy to ensure that we have discovered active remote neighbors by connecting to both servents at the same time and issuing a query with a TTL of three which can only be answered by the crawler servent. The problem with this approach is that connecting to two servents at the same time is problematic due to the restrictions on the neighborhood size of each servent.

Our active approach consists of multiple client servents and a manager. The manager controls the clients in that it supplies each client with a Gnutella servent address (IP address/port number combination) to connect to. Should a client not respond within a reasonable time frame it is restarted. Each client tries to connect to its assigned servent. Depending on success, connection refusal, connection timeout, or Gnutella error message the client reports a different result to the manager. Based on this the manager reschedules the servent for retry. If the connection is rejected with a Gnutella error code it is indicative of an active servent that most likely has no open connection slots currently available. If the connection times out, the servent is either inactive or behind a firewall. If the connection is refused, it is either inactive or highly overloaded with connection requests. Accordingly servents that rejected connections are retried faster than those that refused them or did not respond.

When interacting with other servents, the client is pretending to be a long-running ultrapeer with an acceptable querying scheme. It processes incoming messages and has a non-intrusive Ping/Pong behavior. For example the client issues query/crawler pings only to those peers that have already responded with a Pong. Pings are issued only to those peers that send one themselves, and at the same rate. This seems to avoid bans. The client uses Query messages with a compiled list of catchwords such as mp3, avi, rar. One can expect queries to yield only a subset of neighbors due to the presence of “free-riders” [3].

Early experiments showed that the behavior of a client can have significant impact on the connection success rate. This has led to several changes that make the client more attractive (e.g., large X-Live-Since times, ultrapeer handshaking) as well as less predictable (e.g., initializing the timers that issue the Query and Ping messages with random values within a certain range).

To better understand the limitations of our approach and the behavior of both client and ultrapeers, we experimented with the prevalent tools in a test bed. It consisted of a small Gnutella network with servents based on GTK-Gnutella, LimeWire, BearShare, and Gnucleus. Interestingly only GTK-Gnutella provides a configuration parameter to elevate it to an ultrapeer. We also observed several compatibility issues. For example while the LimeWire servent allows other servents to establish TCP connections to it, it then rejects the Gnutella handshake with an error message. BearShare also discourages other vendors’ servents from connecting to it. We conclude that non-compliance and compatibility issues impose limitations on the success rate of our techniques.

Our passive approach consists of an ultrapeer on the basis of GTK-Gnutella [4]. The goal is to have an ultrapeer that is just a normal node in the network, yet worthwhile to connect to. The ultrapeer shares a reasonable amount of data (100 randomly generated music files, totalling 300 MB in size), and maintains a maximum of 60 simultaneous connections to other servents. To derive various statistics the servent is instrumented to log per-connection information which is augmented with a packet level trace.

Our combined active/passive approach integrates the crawler into the ultrapeer. Experiments with

1In our experiments, see Section Results, the median connection duration is 0.74/0.98 seconds respectively.
the unmodified and the modified ultrapeer confirmed that the changes did not alter the characteristics of incoming connections. Overall this allows us to reach a connection rate well above other known studies (e.g. [5]) during the same time frame.

Results

Our initial characterization is based on a data set collected using the technique discussed in the last section. The trace started on Oct 26, 2003 and lasted till Dec 3, 2003. During this time the ultrapeer logged 8,199,643 sessions of which 8,192,461 are incoming and 7,182 are outgoing. The dominance of the incoming connections indicates that the ultrapeer is quite popular, which is likely to reduce the bias in the sampled servents. The crawler discovered 14,101,399 remote neighbor servents.

Before exploring similarities of the P2P topology with the Internet topology we explore the variability of the Gnutella session durations. Figure 1(a) shows the complementary cumulative distribution function (CCDF) of the session duration of the above trace. It is apparent from the plot that most session durations are rather short. Indeed the median duration of incoming/outgoing sessions is 0.98/0.74 seconds. Only 5% of the incoming sessions lasted longer than 12.3 seconds. This implies that edges in the Gnutella network change rapidly. On the one hand this complicates any crawling attempts, on the other hand it affects the expected accuracy and value of any derived map.

Typical metrics for distances in the Internet are router hop counts and AS distances. Unfortunately, estimating the hop counts for any two random nodes is non-trivial. While difficult, estimating approximate AS distances is possible. We map IP addresses to AS numbers using BGP tables from Ripe [6]. Using BGP tables and updates we derive an AS topology and the AS relationships [7]. Based on this topology and the heuristic that a customer route is preferred to a peering route over an upstream, we estimate the AS distances. Figure 1(b) (solid line) shows a histogram of the estimated AS distances of the remote neighbor servents. The plot shows that the distances span a huge range with some clustering at distance 3 – 5. We note that the estimated AS distances for the direct neighbors have a significantly different distribution. The large values as well as the spread of AS values indicates that Gnutella does not bias its neighbor choices to correspond to network proximity. To further explore this the same Figure (dashed line) shows a histogram of the estimated AS distances of randomly chosen IP addresses. While the overall shape is quite similar there are some differences. This has to be expected since users need reasonable network connectivity to use the Gnutella network.
Summary
Exploring the Gnutella network topology is limited by the optimizations to the Gnutella protocol as well as
the short session durations. Nevertheless we are able to identify a significant number of remote neighbor
servents to approximate a representative set of edges in the P2P network. Our comparison to randomly se-
lected pairs of IP addresses shows that neighbors in the Gnutella P2P network do not seem to significantly
bias their neighbor choices towards network proximity.

References
http://www9.limewire.com/developer/gnutella_protocol_0.4.pdf
http://www.limewire.com
http://www.firstmonday.dk/issues/issue5_10/adar
http://www.gtk-gnutella.sourceforge.net
port, College of Computing, Georgia Tech, 2002
http://data.ris.ripe.net
posium, 2000
Platforms for Simulations of Complex and Mobile Ad Hoc Networks

Giannis Grammatikopoulos¹, Stefan Rührup², Christian Schindelhauer², and Klaus Volbert²

¹ Computer Technology Insitute, Patras   and   University of Patras, Greece
grammat@ceid.upatras.gr

² University of Paderborn, Germany
{sr,schindel,kvolbert}@uni-paderborn.de

In recent scientific literature, much effort has been devoted to defining and analyzing mathematical models, suitable for representing the structure and evolvement of a large variety of real world communication networks, such as the Internet, the World Wide Web or mobile ad hoc sensor networks. Such networks are highly complex and dynamic, since they grow in time with fast rates according to self-organization principles and rules, whose understanding is at least as important as the study of the static properties of their structure.

Within the existing mathematical models, network evolution is usually simulated by means of describing various stochastic processes, through which nodes and edges are inserted or deleted in and from the network graph. None of these models is likely to become universally accepted, since each one necessarily focuses on capturing only a fraction of the features of the real-world systems, while missing some others and introducing unwanted properties and side-effects. As a result, computer simulations and experiments are extremely important in order to evaluate each new model proposal, suitably adjust and improve its definition, and reveal its implications to the networks’ properties.

Large scale experiments are also essential to reveal phase transition phenomena related to the network future growth, prior to attempting rigorous analysis of such phenomena. Setting such experiments aims to measure basic statistical properties of the models under study, such as degree distribution, clustering coefficient, connectedness, diameter etc, but also assists to testing the efficiency of basic functionalities of these networks, such as navigation or resilience to random failures and attacks.

Mainly, we are working together on two platforms: In part of WP 2.1 we develop a simulation platform for complex networks and in part of WP 2.4 we implement new extensions in our simulation environment SAHNE and integrate new testing tools for special problems. In the following, we will describe both activities.

Platform for Complex Networks. In the framework of WP2.1, a unified platform-based software application for simulating and monitoring models for complex networks is under development, with a prototype tool already prepared. Its architecture consists of three parts:

- A basic set of scientific libraries performing simple operations, such as inserting/deleting nodes and edges, changing costs and capacities, making random choices based on various criteria etc. These core functionalities are implemented in C++, using the LEDA collection of graph types and algorithms. The main objective of the platform’s architectural design is, to assemble a powerful set of basic operations which can be combined in various ways to provide simulators for a large class of models’ definitions.

- A user interface for setting and adjusting experiments, as well as for importing and exporting information about the graph models in a convenient way.

- A prototype visualization tool for viewing evolution of networks in time.

At present state of design the simulation platform is capable of simulating models which fall roughly in one or several of the following categories:
• Models based on preferential attachment mechanisms. Preferential attachment is about the assumption that the likelihood of every node receiving new edges is proportional to the node’s degree (number of existing connections).

• Models who take into account evolutionary considerations such as nodes’ fitness, initial attractiveness etc.

• Models which rely on the network geometry (the placement and distances of various nodes in space), extremely relevant to wireless telecommunication systems.

• Models which are based on bipartite graph structures (such as models for social networks representing interactions between actors of two distinct kinds).

The software libraries under development are extending and complementing previous work done in the framework of COSIN IST/FET Project.

**Simulation Environment for Mobile Ad Hoc Networks.** In the framework of WP 2.4, we focus on mobile ad hoc networks (MANETs) and especially on the development of large-scale testing environments for such dynamic and mobile networks. For the experiments we make use of SAHNE, our simulation environment for mobile ad hoc networks [3]. We use SAHNE to close the gap between theoretical investigations and realistic wireless environments. SAHNE has been designed with respect to an ISO/OSI reference model and we use it to simulate mobile ad hoc networks on a level of abstraction that lies between theoretical assumptions and realistic wireless environments. Extensions of SAHNE will help to scale down the gap between theoretical investigations of wireless network topologies and realistic wireless environments. As a first step, we implemented realistic transmission models into SAHNE. Based on this, we can simulate and test algorithms which build up basic network topologies for wireless networks under different transmission models.

![Image](image_url)

*Figure 1: Simulation Environment and Testing Tools for Mobile Ad Hoc Networks*

The new requirements within this project made it necessary to work on further extensions of SAHNE. We have already started implementation of network protocols for MANETs and work on extensions that make it possible to use the same source code in SAHNE as well as on hardware devices, e.g., on a small mini robot. For this purpose we have developed a packet processing library and an interface between this library and SAHNE. The idea behind it is that we can evaluate algorithms in SAHNE and that it is not necessary to re-implement useful algorithms on the mini robot.

In our approach we model MANETs by geometric graphs. Recently, we investigated important measures like congestion, dilation and energy for point-to-point communication problems in wireless networks [1] and analyzed the relation between different types of spanners [4].

We have already implemented distributed algorithms into SAHNE that build up basic network topologies for mobile ad hoc networks. In a next step, we want to integrate distributed algorithms for MANETs.
for moving radio stations (see also [1]) with adjustable transmission power in a worst case scenario. We introduced two worst-case models for node movements and presented first algorithms that can maintain persistent routes in a network with nice communication network properties like hop-distance, energy-consumption, congestion and number of interferences. One aim is to implement these algorithms into SAHNE and to investigate their behaviour experimentally. In addition, we want to integrate network protocols developed and analyzed in [2] for position-based routing into SAHNE.

Besides simulations with SAHNE we have also implemented another testing environment in Java to investigate elementary graph properties of sectorized topologies [6]. We could use this tool already to analyze adjustable, sectorized topologies for static ad hoc networks experimentally [5].

References


A new Infrastructure for the Management of Networks and Grids based on Peer-to-Peer

Giovanni Cortese¹, Felix Heine², Federico Morabito¹, and Fabrizio Davide¹

¹ Telecom Italia Learning Services, Rome, Italy
g.cortese@computer.org
{Fabrizio.Davide,Federico.Morabito}@telecomitalia.it
² University of Paderborn, Germany
fh@upb.de

A joint work by TILS and UPB within Subproject 2 is targeting the design and implementation of an information management architecture which should serve the need both of the network, systems and application management domain (we deliberately include this based on our belief that a strictly layered approach will tend to disappear in future IT infrastructures) and the management of Grids.

In the management domain, we support a vision where information will no longer be exchanged only along a manager-agent hierarchy, as it is today based on the SNMP and Policy architectures defined by IETF, which represent the standards for monitoring and configuration of networks, respectively. Instead, we build on communicating in a peer-to-peer fashion among nodes. A new monitoring architecture should then support a scalable, data centric communication model, allowing efficient many-to-many information exchange. Also, the information model should be much more flexible than today, allowing peers to communicate even when data schemas have not been agreed in advance as in the currently used standard-based approach.

Management of Grids shares similar challenges with network management (NM). Future Grids will be much larger than current Grids, leading to the vision of a world-scale Grid which might be the basis for the next generation of the Internet. These Grids will be highly dynamic and very heterogeneous. Thus, the same requirements for an information model arise, e.g. a model to describe the resources of the Grid. We use ontologies as a flexible way to describe the information model. Additionally, we use Description Logics (DL) to be able to deduce information from these ontologies.

Regarding the aspect of Grid management, we focus on the problem of resource discovery. As resource types and background knowledge about resources constantly evolves, no single standard for resource description can exist. The management platform will have to cope with multiple ontologies which are maintained locally.

To this end we are developing an information management architecture targeting large, dynamic environments, and we are providing an implementation (which we will test soon on PlanetLab), which enables peers to:

- Exchange management information, specifically
  - share configuration information (that is, moderately static, persistent data describing components of the infrastructure), in order to support advanced resource and information discovery capabilities. For Grids, this would be targeted to allow computing nodes to publish information about the resources they offer. In management, this feature would again support realization of a resource directory, but also support fine grained discovery of nodes supporting specific 'MIBs' (that is, nodes that could be provider of certain types of information)
  - share 'observations' i.e. data streams generated from 'sensors' within the infrastructure, related to properties of the system that change frequently with time. This is the central task
for NM, where collection of large bodies of accounting and performance data is a fundamental requirement; for Grids it allows resources to publish current levels of utilizations, usage histories, performance scores, etc.

- Maintain a distributed ontology, which provides the structure of the management information. Our long-term goal is to also provide reasoning capabilities over the distributed ontology.

Our design leverages distributed hash table (DHT) overlay networks. On top of a DHT routing overlay, we provide a pull-style query service, and a notification-based service. Additionally, we use DHT networks to generate a distributed hierarchy of concepts which integrates concepts from multiple locally defined ontologies. By this, we build a consistent view spanning multiple local schemas. This can, e.g. be used to discover resources described by concepts which are unknown to the local provider.

At the current state of the art, two distinct prototypes have been realized, focusing on storage and index management issues for configuration and large volume observation streams, and one implementing the generation of a distributed hierarchy of concepts from multiple ontologies, respectively. We are currently working at an integrated design which we envision to provide for the completion of first phase of DELIS.

The prototype for storage and index management is composed by three main components:

- Content-based publish-subscribe, to allow clients to register for management information of interest and to collect and distribute management information.
- Distributed Persistent Data Storage, with appropriate support for indexing and complex query mechanisms. Data confidentiality and privacy are also supported using cryptographic algorithms.
- Query processor, supporting XML Xpath and indexing mechanism.

The prototype for the distribution of ontology concepts features the following:

- Use a local DL reasoner at each node to generate a local hierarchy.
- Use a DHT-based network to build a distributed hierarchy of all concepts from all nodes, and propagate instances along this hierarchy.
- Do simple instance retrieval over the distributed information.

References

Typical Properties of Winners and Losers in Discrete Optimization

Rene Beier and Berthold Vöcking

1 Max Planck Institut für Informatik, Saarbrücken, Germany and ICSI Berkeley, USA
rbieier@mpi-sb.mpg.de

2 RWTH Aachen, Germany
voiecking@i1.informatik.rwth-aachen.de

Many combinatorial optimization problems have an objective function or constraints specified in terms of real numbers representing natural quantities like time, weight, distance, or utility. This includes some well-studied large-scale optimization problems from network design and routing like, e.g., traveling salesman, shortest path, minimum spanning tree as well as various scheduling and packing problems. When analyzing the complexity of algorithms for such problems, we usually assume that these numbers are integers or rational numbers with a finite length representation. The hope is that it suffices to measure and compute with some bounded precision in order to identify an optimal or close to optimal solution. In fact, if real numbers occur only in the objective function and if this objective function is well-behaved (e.g., a linear function) then calculating with reasonable approximations of the input numbers yields a feasible solution whose objective value is at least close to the optimal objective value. More problematically, however, if the constraints are defined by real numbers, then calculating with rounded input numbers might miss all interesting solutions or might even produce infeasible solutions.

How can one solve optimization problems (efficiently) on a computer when not even the input numbers can be specified exactly? – In practice, optimization problems in which real numbers occur in the input are solved by simply rounding the real numbers more or less carefully. Fortunately, this approach seems to yield reasonable results. We seek for a theoretically founded explanation why this rounding approach usually works. Studying this issue under worst case assumptions does not make very much sense as, in the worst case, the smallest inaccuracy might lead to an infeasible or utterly sub-optimal solution. This question needs to be studied in a stochastic model. We present a probabilistic analysis for a very large class of combinatorial optimization problems containing, e.g., all binary optimization problems defined by linear constraints and a linear objective function over \{0, 1\}^n, but also problems with a non-linear feasible region. By parameterizing which constraints are of stochastic and which are of adversarial nature, we obtain a semi-random input model that enables us to do a general average-case analysis for a large class of optimization problems while at the same time taking care for the combinatorial structure of individual problems. Our analysis covers various probability distributions for the choice of the stochastic numbers and includes smoothed analysis with Gaussian and other kinds of perturbation models as a special case. For example, the feasible region can be described by an arbitrary set \(S \subseteq \{0, 1\}^n\) intersected by some stochastic constraints, where the set \(S\) is assumed to be specified by an adversary in an arbitrary possibly non-linear and non-convex way and only the stochastic constraints are assumed to be linear. In this semi-random input model, we can exactly characterize the smoothed complexity of optimization problems in terms of their random worst-case complexity.

A binary optimization problem has a polynomial smoothed complexity if and only if it has a pseudopolynomial complexity.

Our analysis is centered around structural properties of binary optimization problems, called winner, loser, and feasibility gaps. We show, when the coefficients of the objective function and/or some of the constraints are stochastic, then there usually exist a polynomial \(n^{-\Omega(1)}\) gap between the best and the second best solution as well as a polynomial slack to the boundary of the constraints. Similar to the condition
number for linear programming, these gaps describe the sensitivity of the optimal solution to slight perturbations of the input and can be used to bound the necessary accuracy as well as the complexity for solving an instance. Our analysis shows that one can usually round real-valued input numbers after only a logarithmic number of bits without changing the optimal solution. In fact, our probabilistic analysis goes far beyond the point of explaining phenomena occurring in practice. We are able to provide algorithms with polynomial average-case complexity (more precisely, polynomial smoothed complexity) for a quite general class of discrete optimization problems. Technically, we exploit the gap properties in form of an adaptive rounding scheme increasing the accuracy of calculation until the optimal solution is found. The strength of our techniques is illustrated by applications to various NP-hard optimization problems from mathematical programming, network design, and scheduling for which we obtain the first algorithms with polynomial average-case/smoothed complexity. For example, we obtain algorithms with polynomial smoothed complexity for the multi-dimensional knapsack problem, the constrained spanning tree problem, the constrained shortest path problem, and the problem of scheduling to minimize the weighted number of tardy jobs.

References

Efficient Smart Dust Protocols

Sotiris Nikoletseas and Paul Spirakis

Computer Technology Institute, Patras and University of Patras, Greece

\{nikole,spirakis\}@cti.gr

Overview

We present several efficient and fault-tolerant protocols for wireless sensor networks, as well as new random graph-theoretic models that may accurately model such networks. Besides the theoretical analysis, we implement and evaluate our protocols using a simulator (SimDust) that we have created. Finally, we describe some relevant wide dissemination activities (new conferences initiated by CTI researchers, book chapters, articles in popular science columns of major newspapers).

Models and Protocols

Smart Dust is comprised of extremely small devices that can sense local conditions, perform computations and communicate wirelessly with each other. Such wireless sensor networks have important applications, like in area monitoring, security, precise agriculture, ambient intelligence etc.

The CTI sensors group, in cooperation with leading groups in Europe and the US, have designed and validated a variety of new protocols that manage to propagate data in such networks in a fast, cheap and robust way.

For the problem of data propagation we have proposed in [3] a new protocol which tries to minimize energy consumption by probabilistically favoring certain paths of local data transmissions towards the sink. Thus, we call this protocol PFR (Probabilistic Forwarding Protocol).

The basic idea behind this protocol is to avoid flooding by favoring (in a probabilistic manner) data propagation along sensors which lie “close” to the (optimal) transmission line, ES, that connects the sensor node detecting the event, E, and the sink, S. This is implemented by locally calculating the angle \( \phi = EPS \), whose corner point \( P \) is the sensor currently running the local protocol, having received a transmission from a nearby sensor, previously possessing the event information. If \( \phi \) is equal or greater to a predetermined threshold, then \( p \) will transmit (and thus propagate the event information further). Else, it decides whether to transmit with probability equal to \( \frac{\phi}{\pi} \).

Because of the probabilistic nature of data propagation decisions and in order to prevent the data propagation process from early failing, we initially use (for a short time period which we evaluate) a flooding mechanism that leads to a sufficiently large “front” of sensors possessing the data under propagation. When such a “front” is created, we perform probabilistic forwarding.

The protocol, as shown by a geometric analysis, always propagates data to the sink, under ideal network conditions (no failures), thus it is provably correct. Using properties of stochastic processes, we show that the protocol is very energy efficient. Also, when part of the network is inoperative (which is more realistic, because sensors are prone to faults), the protocol manages to propagate data very close to the sink, thus it is robust. The above analytical results are validated by large scale experiments we have carried out after implementing the protocol.

In [5] we study the problem of energy-balanced data propagation in wireless sensor networks. The energy balance property guarantees that the average per sensor energy dissipation is the same for all sensors, during the entire execution of the protocol. This property is important since it prolongs the network’s lifetime by avoiding early energy depletion of sensors.
We propose a new algorithm that in each step decides whether to propagate data one-hop towards the final destination (the sink), or to send data directly to the sink. This randomized choice balances the (cheap) one-hop transmissions with the direct transmissions to the sink, which are more expensive but “bypass” the sensors lying close to the sink. Note that, in most protocols (including the PFR protocol presented above), these close to the sink sensors tend to be overused and die out early.

By a detailed analysis we precisely estimate the probabilities for each propagation choice in order to guarantee energy balance. The needed estimation can easily be performed by current sensors using simple to obtain information. Under some assumptions, we also derive a closed form for these probabilities.

The fact (shown by our analysis) that direct (expensive) transmissions to the sink are needed only rarely, shows that our protocol, besides energy-balanced, is also energy efficient.

In [6] we propose new random graph-theoretic model that may serve as abstract models for wireless sensor networks. In particular, we propose a new model for random intersection graphs \((G_{n,m,p})\).

In random intersection graphs, to each of the \(n\) vertices of the graph, a random subset of a universal set of \(m\) elements is assigned, by independently choosing elements with the same probability \(p\). Two vertices \(u, v\) are then adjacent in the \(G_{n,m,p}\) graph if and only if their assigned sets of elements have at least one element in common. We extend this model (which we call hereafter “uniform”, because of the same probability of selecting elements) by proposing two new models.

Random intersection graphs may model real-life applications more accurately (compared to the \(G_{n,p}\) case). This is because in many cases the independence of edges is not well-justified. In fact, objects that are closer (like moving hosts in mobile networks or sensors in smart dust networks) are more probable to interact with each other.

We also investigate the existence and efficient algorithmic construction of close to optimal independent sets in random models of intersection graphs. For this model we derive exact formulae for the mean and variance of the number of independent sets of size \(k\) (for any \(k\)) in the graph. We then propose and analyse three algorithms for the efficient construction of large independent sets in this model. The first two are variations of the greedy technique while the third is a totally new algorithm. Our algorithms are analysed for the special case of uniform random intersection graphs. Our analyses show that these algorithms succeed in finding close to optimal independent sets for an interesting range of graph parameters.

**SimDust: A Simulator for Smart Dust**

Besides the mathematical analysis, these protocols have been implemented in software developed locally (SimDust, see [1]) and evaluated in detail under practical conditions. **SimDust** was implemented in *Linux* using *C++* and the LEDA algorithmic and data structures library.

An interesting feature of our simulator, is the ability to experiment with very large networks. In fact, the complexity of extending existing networks simulators, and their (in cases of large instances) time consuming execution, were two major reasons for creating this simulator. **SimDust** enables the protocol designer to implement the protocol using just *C++* and avoids complicate procedures that involve the use of more than one programming language.

Additionally, **SimDust** generates all the necessary statistics at the end of each simulation based on a big variety of metrics that are implemented (such as delivery percentage, energy consumption, delivery delay, longlivety, etc.)

Some key points in **SimDust**’s implementation are the following:

**Operation in rounds:** A basic concept used in the simulator is that its operation is divided into discrete rounds. One round represents a time interval in which a particle can transmit or receive a message and process it according to the protocol that is being simulated.

**Energy assumptions:** We have included a detailed energy dissipation scheme. In particular, we have assumed that a particle consumes a standard amount of energy \(E_{elec}\) per round while being awake. Furthermore, in each transmission energy fading is proportional to the square of the distance. For each receive, a node is credited with an amount of energy that practically reflects the power needed to run the...
transceiver circuit namely \( E_{\text{elec}} \). Finally, a particle can switch over to the sleep state, to save energy. No energy consumption virtually takes place while the particle remains in the sleep mode, since it keeps its transceiver and its sensors shut down.

We also study collision avoidance problems in wireless sensor networks and provide adaptive protocols in [4].

Also, some basic functions and primitives have been implemented in real smart dust systems, like the MRB "motes" developed at Berkeley that run a small but efficient operating system (TinyOS).

**Wide Dissemination Activities**

Furthemore, the CTI people have contributed to creating new, focused relevant events, like ALGOSENSORS 04 (“Algorithmic Aspects of Wireless Sensor Networks”, http://ru1.cti.gr/algosensors04/, [7], that was held in Finland) and DCOSS 05 (“Distributed Computing in Sensor Systems”, http://www.dcos.org/, to be held in California). An article of Paul Spirakis in [7] poses critical challenges of the future smart dust research [8]. Also, towards widely disseminating this activity, several chapters in books of international circulation have been published (see [2]), and also articles in popular science columns of major newspapers have appeared.

**References**


A new approach to compute schedules for unrelated parallel machines

Martin Gairing¹, Thomas Lücking¹, Burkhard Monien¹, Marios Mavronicolas², and Andreas Woclaw¹

¹ University of Paderborn, Germany
{gairing,luck,bm,wocland}@upb.de

² University of Cyprus
mavronic@cs.ucy.ac.cy

Scheduling unrelated parallel machines is among the most difficult scheduling problems. Independent jobs have to be assigned to a set of machines of different type without preemption. Processing job \( i \) on machine \( j \) takes time \( p_{ij} \). For each machine \( j \), the total time used by machine \( j \) is the sum of processing times \( p_{ij} \) for the jobs that are assigned to machine \( j \). The makespan of a schedule is the maximum total time used by any machine. The objective is to find a schedule (assignment) that minimizes the makespan.

The unrelated scheduling problem is \( \mathcal{NP} \)-complete, and it is known that unless \( \mathcal{P} = \mathcal{NP} \), there is no polynomial-time approximation algorithm for the optimum schedule with approximation factor less than \( \frac{3}{2} \). There exist polynomial-time algorithms with an approximation factor of 2. These algorithms first compute an optimal fractional assignment and then use rounding to generate a discrete solution. The optimal fractional assignment can be computed by solving an LP (using interior point methods guarantees polynomial time), or combinatorially with the help of generalized network flows.

Many real-life large-scale optimization problems can be modeled as an unrelated scheduling problem. Typically, these problems arise in the area of scheduling multiprocessor computers and industrial manufacturing systems, where often groups of similar workstations having a variety of similar equipment with differing performance characteristics have to be optimized. For practical purposes, there have been proposed many heuristics solving the unrelated scheduling problem. They deliver both optimal and approximation solutions. Techniques used here range from combinatorial approaches with partial enumeration to integer programming with branch-and-bound and cutting planes.

The algorithm, we present in [2], computes an assignment for the unrelated scheduling problem with makespan at most twice the optimum. We prove that a 2-approximative schedule can be computed in \( O(m^2A\log(m\log(nU))) \) time, where \( A \) is the number of pairs \( (i,j) \) with \( p_{ij} \neq \infty \) and \( U \) denotes the maximal \( p_{ij} \neq \infty \). This is better than the previously known best time bounds. Our algorithm is linear in \( A \). This is the first time that a combinatorial algorithm always beats the interior point approach for this problem. For problem instances where \( A = \Theta(nm) \) our algorithm outperforms best known approaches by a factor almost linear in \( n \).

Our algorithm is much simpler than the previously known algorithms. It always maintains an integral solution. It pushes unsplittable jobs through a two-layered bipartite network defined by the unrelated scheduling problem. It uses the primal-dual approach combined with a gain scaling technique to obtain a polynomial running time. To compute a flow among the edges with zero reduced cost it uses the procedure Unsplittable-Blocking-Flow from [1] in the inner loop.

A special case of the unrelated scheduling problem is the restricted scheduling problem, in which each job \( i \) has some weight \( w_i \), each machine \( j \) has some speed \( s_j \) and \( p_{ij} = \frac{w_i}{s_j} \) or \( p_{ij} = \infty \) holds for all \( i, j \). In [1], we design the basic procedure Unsplittable-Blocking-Flow and we give a combinatorial 2-approximation algorithm for the restricted scheduling problem. Our solution is based on the generic preflow-push algorithm. Moreover, for a simpler case with identical machines, our algorithm delivers an assignment at most \( 2 - \frac{1}{\hat{w}_1} \) times worse than the optimal solution, where \( \hat{w}_1 \) is the largest job weight. It improves on the best known approximation factor for this job scheduling problem.
Furthermore, for the restricted scheduling problem with identical machines, we present an algorithm which computes a pure Nash equilibrium from an arbitrary (not necessarily equilibrium) assignment without increasing the makespan. This technique is called Nashification. The algorithm combines the techniques from blocking flows with unsplittable flows. This is the first polynomial-time algorithm to compute a Nash equilibrium for this problem.

We feel that our approach has led to a much better understanding of the structure of the unrelated scheduling problem. A mixed continuous/discrete approach has been replaced by a completely discrete approach. Our algorithm takes advantage from addressing the approximation problem directly.

References


Complex networks, such as the Internet, consist of multiple self-interested agents (or users) with varying degrees of collaborative motives. A fundamental problem in this context, that we study in this work, is the design of mechanisms that share the cost of deploying a network infrastructure in a fair manner among the participating agents such that agents are incited to cooperate. A preliminary version of this work will appear in [4].

Mechanism design and approximation algorithms. The general framework that we consider is as follows. We are given a set $R$ of agents that wish to receive a service, e.g., being connected to a common network infrastructure. For a subset $Q \subseteq R$ the cost of servicing all agents in $Q$ is denoted by $c(Q)$. Each agent $j \in R$ has a private utility $u_j$, which is the maximum price she is willing to pay for this service. A cost sharing mechanism elicits from each agent $j \in R$ a bid $b_j$ for the service and, based on these bids, determines a set $Q \subseteq R$ of agents that receive service and how the servicing costs $c(Q)$ are distributed among the agents in $Q$. Let $x_j$ be the cost share of $j \in Q$ computed by the mechanism. A cost sharing mechanism is budget balanced if the agents in $Q$ pay exactly the cost $c(Q)$, i.e., $\sum_{j \in Q} x_j = c(Q)$.

We define the benefit of an agent $j \in R$ to be $u_j - x_j$ if $j \in Q$ and 0 otherwise. The decisions made by the mechanism are based solely on the bids received from the agents. Since each agent is selfish, she may lie about her true utility so as to maximize her benefit. We are interested in incentive compatible cost sharing mechanisms, i.e., mechanisms that encourage agents to bid their true utility. There are two notions in this context: A cost sharing mechanism is strategyproof if no agent has an incentive to lie about her true utility; it is said to be group-strategyproof if no coalition of agents has an incentive to jointly misreport their true utility.

Moulin and Shenker [18] show that budget balanced cost sharing mechanisms that achieve group-strategyproofness can be obtained from budget balanced cost sharing methods that are cross-monotonic. A cost sharing method $\xi$ is a function that, given any subset $Q \subseteq R$ of agents, assigns a non-negative cost share $\xi_j(Q)$ to each agent $j \in Q$. $\xi$ is said to be cross-monotonic if the cost share of each individual agent never increases as the set of participating agents grows; more formally, $\forall P \subseteq Q, \forall j \in P, \xi_j(P) \geq \xi_j(Q)$. Intuitively this is a very natural notion. As more agents start receiving service, the fraction of that service a single agent has to pay cannot increase.

Ideally we would like the cost sharing mechanism to be budget balanced; however, here we are interested in mechanisms that are computable in polynomial time and thus we cannot enforce this condition if the underlying optimization problem is NP-hard. Jain and Vazirani [3] extended the result of Moulin and Shenker [18] to approximately budget balanced cost sharing methods. A cost sharing method $\xi$ is $\alpha$-budget balanced if

$$\frac{1}{\alpha} \cdot c(Q) \leq \sum_{j \in Q} \xi_j(Q) \leq \text{opt}(Q),$$

where $c(Q)$ and opt$(Q)$ denote the cost of an approximate and optimal solution to service all agents in $Q$, respectively. The intuition behind this is that the agents will never pay more than the optimum cost
opt(Q), but at least a 1/α fraction of the cost of the computed solution is recovered. Following the ideas in [18], Jain and Vazirani prove that an α-budget balanced and cross-monotonic cost sharing method gives rise to an α-budget balanced group-strategyproof cost sharing mechanism.

Our contribution and related work. We present a 2-budget balanced cross-monotonic cost sharing method for the Steiner forest game. An instance of this game consists of an undirected graph G = (V,E), a non-negative cost function c : E → R+ on the edges of G, and a set R ⊆ V × V of k terminal pairs. Each terminal pair (s,t) ∈ R is associated with an agent that wishes to establish a connection between vertices s and t in the underlying network. A feasible solution is a forest F that contains an s,t-path for each connection request (s,t) ∈ R.

Previously, Jain and Vazirani [3] gave a 2-budget balanced and cross-monotonic cost sharing method for the Steiner tree game — a special case of the game considered here. Such a result for Steiner forest games has proved to be elusive so far, in stark contrast to the well known primal-dual (2 − 1/k)-approximation algorithms [1, 2] for the problem. In a recent work, van Zwam [6] proved that no cross-monotonic cost sharing method for Steiner trees is α-budget balanced with α < 2. This implies that our result is tight.

Our algorithm is an original extension of the known primal-dual algorithm for Steiner forests [1]. An interesting byproduct of our work is that our Steiner forest algorithm is (2 − 1/k)-approximate despite the fact that the forest computed by our method is usually costlier than those computed by known primal-dual algorithms. In fact the dual solution computed by our algorithm is infeasible for the classical undirected cut relaxation for Steiner forests but we can still prove that its total value is at most the cost of a minimum-cost Steiner forest for the given instance. Furthermore, our algorithm gives rise to a new undirected cut dual for Steiner forests, which we coin the lifted-cut dual. We show that this new dual is strictly stronger than the standard undirected cut dual for Steiner forests.

LP formulations and primal-dual algorithms. We review the standard integer LP formulation for the Steiner forest problem. We have a variable xe for each edge e ∈ E which has value 1 if e is part of the resulting forest and 0 otherwise. A subset U ⊆ V is called a Steiner cut if it separates at least one terminal pair in R, i.e., there exists a pair (s,t) ∈ R such that |{s,t} ∩ U| = 1. We let Ω be the set of all Steiner cuts. We also define δ(U) as the set of edges that have exactly one endpoint in U.

\[
\min \sum_{e \in E} c_e x_e \quad \text{s.t.} \quad \sum_{e \in \delta(U)} x_e \geq 1 \quad \forall U \in \Omega, \text{ } x \text{ integer.} \quad \text{(IP)}
\]

The linear program dual of the LP-relaxation (LP) of (IP) has a variable yU for all vertex sets U ∈ Ω. There is a constraint for each edge e ∈ E that limits the total dual assigned to sets U ∈ Ω that contain exactly one endpoint of e to be at most ce:

\[
\max \sum_{U \in \Omega} y_U \quad \text{s.t.} \quad \sum_{U \in \Omega : x \in \delta(U)} y_U \leq c_e \quad \forall e \in E, \text{ } y \geq 0. \quad \text{(D)}
\]

The algorithm AKR due to Agrawal, Klein, and Ravi [1] is a primal-dual algorithm; that is, it constructs both a feasible and integral primal solution to (LP) and a feasible dual solution to (D). We can think of an execution of AKR as a process over time. Let x' and y' be the primal incidence vector and feasible dual solution at time t. We use F' to denote the forest corresponding to x'. Initially, x'_e = 0 for all e ∈ E and y'_U = 0 for U ∈ Ω. In the following we say that an edge e ∈ E is tight if the corresponding constraint in (D) holds with equality. Assume that the forest F' at time t is infeasible. We use F'' to denote the subgraph of G that is induced by the tight edges for dual y'. A connected component U of F'' is active iff U separates at least one terminal pair, i.e., iff U ∈ Ω. Let A' be the set of all active connected components of F'' at time t. AKR raises the dual variables for all sets in A' uniformly at all times t ≥ 0. Suppose now that two active connected components U_1 and U_2 collide at time t in the execution of AKR. In other words, there are terminals u ∈ U_1 and v ∈ U_2 such that a path between u and v becomes tight as a consequence
of increasing $y_{U_1}$ and $y_{U_2}$. If this happens, we add the path to $F'$ and continue. $U_1$ and $U_2$ are part of the same connected component of $F'$ for $t' > t$.

Suppose that AKR outputs a forest $F$ and a feasible dual solution $\{y_U\}_{U \in \mathcal{U}}$. Agrawal, Klein, and Ravi [1] proved that $c(F) \leq (2 - 1/k) \cdot \sum_{U \in \mathcal{U}} y_U \leq (2 - 1/k) \cdot \text{opt}(R)$.

A cross-monotonic cost sharing algorithm for Steiner forests. We next describe the modifications that are necessary to turn AKR into a cross-monotonic cost sharing algorithm. We use CSF to refer to this algorithm.

Define the time of death $d(s,t)$ for each terminal pair $(s,t) \in R$ as

$$d(s,t) := \frac{1}{2} \cdot c(s,t),$$

where $c(s,t)$ denotes the cost of the minimum-cost $s,t$-path in $G$. We assume for ease of presentation that each vertex $v \in V$ has at most one terminal on it. This assumption is without loss of generality since we can replace each vertex in $V$ by a sufficient number of copies and link these copies by zero-cost edges. We extend the death time notion to individual terminals and define $d(s) = d(t) = d(s,t)$ for terminals $s,t \in R$.

Using the notation introduced above we obtain CSF by modifying the definition of $\mathcal{A}$ and $\mathcal{A}'$. We say that a connected component $U$ of $F'$ is active at time $t$ if it contains at least one terminal $v \in U$ with death time at least $t$, i.e., $U$ is active iff there exists a $v \in U$ with $d(v) \geq t$. CSF grows all active connected components in $\mathcal{A}'$ uniformly at all times $t \geq 0$. Observe that this way CSF also raises dual variables of connected components in $\mathcal{A}'$ that do not correspond to Steiner cuts. In what follows we denote by $\mathcal{U}$ the set of all Steiner and non-Steiner cuts raised during the execution of CSF. The intuition behind CSF is that a terminal pair $(s,t)$ is active for the time it would take $s$ and $t$ to connect in the absence of all other terminals. Therefore its activity time is independent of other terminal pairs and this is crucial to achieve cross-monotonicity.

For a terminal $v \in R$ and for $t \leq d(v)$ we let $U'(v)$ be the connected component in $F'$ that contains $v$. Also let $d'(v)$ be the number of terminals in $U'(v)$ whose death time is at least $t$. The cost share of a terminal vertex $v \in R$ is defined as

$$\xi_v(R) := \int_{t=0}^{d(v)} \frac{1}{d'(v)} \, dt.$$ 

Furthermore, we define $\xi_{(s,t)}(R) := \xi_s(R) + \xi_t(R)$ for all $(s,t) \in R$. The following is our main theorem:

**Theorem** $\xi$ is a cross-monotonic cost sharing method that is 2-budget balanced.

One of the major difficulties to prove the above theorem is to show that the sum of the cost shares is at most the optimal cost $\text{opt}(R)$. Since we share the entire dual produced during the execution of CSF among the terminal pairs in $R$, this claim would follow immediately from weak duality if $\{y_U\}_{U \in \mathcal{U}}$ were a feasible solution for (D). However, CSF also raises dual variables of non-Steiner cuts and thus we cannot apply this argument here. This raises the question of whether there is an alternate Steiner forest LP formulation such that the cost shares computed by CSF correspond to a feasible dual solution. In fact, CSF gives rise to a new lifted-cut dual formulation and we can prove that this new formulation is strictly stronger than the standard LP dual (D). This raises the dazzling question of the existence of better primal-dual algorithms for Steiner trees and forests.

**References**


Selfish Unsplittable Flows

Dimitris Fotakis¹, Spyros Kontogiannis², and Paul Spirakis²

¹ Aristotle University of Thessaloniki, Greece
fotakis@gen.auth.gr

² Computer Technology Institute, Patras, Greece
{kontog,spirakis}@cti.gr

Consider a model where selfish users having varying demands compete for some shared resources. The quality of service provided by a resource decreases with its congestion, i.e., the amount of demands of the users willing to be served by it. Each user may reveal its actual (unique) choice (called a pure strategy) among the resources available to it, or it may reveal a probability distribution for choosing one of its candidate resources (a mixed strategy). The users determine their actual behavior based on other users’ behavior, but they do not cooperate. We are interested in situations where the users have reached some kind of equilibrium. The most popular notion of equilibrium in non-cooperative game theory is the Nash equilibrium: a “stable point” among the users, from which no user is willing to deviate unilaterally. In [15] the notion of the coordination ratio or price of anarchy was introduced, as a means for measuring the performance degradation due to lack of users’ coordination when sharing common goods.

A realistic scenario for the above model is when unsplittable demands are routed selfishly in general networks with load-dependent edge delays. When the underlying network consists of two nodes and parallel links between them, there has been an extensive study on the existence and computability of equilibria, as well as on the price of anarchy. Motivated by the work of [15], we generalize their concept to the weighted congestion games in a non-trivial way. In this kind of games we study complexity issues concerning the existence and construction of pure Nash equilibria (i.e., Nash equilibria in which all the users adopt pure strategies), as well as the price of anarchy for some particular instances of these games.

The Model. Consider having a set of resources $E$ in a system. For each $e \in E$, let $d_e(\cdot)$ be the delay per user that requests its service, as a (non-decreasing) function of the total usage (i.e., congestion) of this resource by all the users. Each resource may be represented by a pair of points: an entry point to the resource and an exit point from it. So, we represent each resource by an arc from its entry point to its exit point and we associate with this arc the cost (e.g., the delay as a function of the load of this resource) that each user has to pay if she is served by this resource. The entry/exit points of the resources need not be unique; they may coincide in order to express the possibility of offering joint service to users, that consists of a sequence of resources. We denote by $V$ the set of all entry/exit points of the resources in the system. Any nonempty collection of resources corresponding to a directed path in $G \equiv (V, E)$ comprises an action in the system.

Let $N \equiv [n]$ be a set of users, each willing to adopt some action in the system. For each $i \in N$, let $w_i$ denote user $i$’s demand (e.g., the flow rate from a source node to a destination node), while $\Pi_i \subseteq 2^E \setminus \emptyset$ is the collection of actions, any of which would satisfy user $i$ (e.g., alternative routes from a source to a destination node, if $G$ represents a communication network). The collection $\Pi_i$ is called the action set of user $i$ and each of its elements contains at least one resource. Any tuple $\sigma \in \Pi \equiv \times_{i=1}^n \Pi_i$ is a pure strategies profile, or a configuration of the users. Any real vector $p = (p_1, p_2, \ldots, p_n)$ s.t. $\forall i \in [n], p_i : \Pi_i \to [0, 1]$ is a probability distribution over the set of allowable actions for user $i$, is called a mixed strategies profile for the $n$ users.

A congestion model typically deals with users of identical demands, and thus, user cost function depending on the number of users adopting each action ([21, 20, 5]). In this work we consider the more general case, where a weighted congestion model is the tuple $((w_i)_{i \in N}, (\Pi_i)_{i \in N}, (d_e)_{e \in E})$. That is, we allow the users to have different service demands from the whole system, and thus affect the resource delay...
functions in a different way, depending on their own weights. The weighted congestion game associated with this model, is the game in strategic form with the set of users $N$ and user demands $(w_i)_{i \in N}$, the action sets $(\Pi_i)_{i \in N}$ and cost functions $(\lambda^i_{\sigma})_{i \in N, \sigma \in \Pi_i}$ defined as follows: For any configuration $\sigma \in \Pi$ and $\forall e \in E$, let $\Lambda_i(\sigma) = \{ i \in N : e \in \sigma_i \}$ be the set of users exploiting resource $e$ according to $\sigma$ (called the view of resource $e$ w.r.t configuration $\sigma$). The cost $\lambda^i(\sigma) = \sum_{e \in \sigma_i} d_e(\hat{\nu}_e(\sigma))$ where, $\forall e \in E, \hat{\nu}_e(\sigma) = \sum_{i \in \Lambda_i(\sigma)} w_i$ is the load on resource $e$ w.r.t the configuration $\sigma$.

On the other hand, for a mixed strategies profile $p$, the expected cost of user $i$ for adopting strategy $\sigma_i \in \Pi_i$ is $\lambda^i_{\sigma_i}(p) = \sum_{i \in \Pi_i} \cdot P(\sigma^{-i}, \sigma_i) \cdot \sum_{e \in \sigma_i} d_e(\hat{\nu}_e(\sigma^{-i} \cup \sigma_i))$ where, $\sigma^{-i}$ is a configuration of all the users except for $i$, $p^{-i}$ is the mixed strategies profile of all users except for $i$, $\sigma^{-i} \cup \sigma_i$ is the new configuration with $i$ choosing strategy $\sigma_i$,

A congestion game in which all users are indistinguishable (i.e., they have the same user cost functions) and have the same action set, is called symmetric. When each user’s action set $\Pi_i$ consists of sets of resources that comprise (simple) paths between a unique origin-destination pair of nodes $(s_i, t_i)$ in $(V,E)$, we refer to a network congestion game. If additionally all origin-destination pairs of the users coincide with a unique pair $(s, t)$ we have a single commodity network congestion game and then all users share exactly the same action set. Observe that a single-commodity network congestion game is not necessarily symmetric because the users may have different demands and thus their cost functions will also differ.

**Selfish behavior.** Fix an arbitrary (mixed in general) strategies profile $p$ for a congestion game $((w_i)_{i \in N}, (\Pi_i)_{i \in N}, (d_e)_{e \in E})$. We say that $p$ is a Nash Equilibrium (NE) if and only if $\forall i \in N, \forall \sigma_i, \pi_i \in \Pi_i, p_i(\sigma_i) > 0 \Rightarrow \lambda^i_{\sigma_i}(p) \leq \lambda^i_{\pi_i}(p)$. A configuration $\sigma_i \in \Pi$ is a Pure Nash Equilibrium (PNE) if and only if $\forall i \in N, \forall \sigma_i \in \Pi_i, \lambda^i_{\sigma_i}(\sigma) \subseteq \lambda^i_{\sigma^{-i} \cup \sigma_i}(\pi_i)$ where, $\sigma^{-i} \cup \sigma_i$ is the same configuration with $\sigma$ except for user $i$ that now chooses action $\pi$. The social cost $\text{SC}(p)$ in this congestion game is $\text{SC}(p) = \sum_{\sigma_i \in \Pi} P(p, \sigma) \cdot \max_{i \in N} \lambda^i_{\sigma_i}(\sigma)$ where $P(p, \sigma) = \prod_{i \in N} p_i(\sigma_i)$ is the probability of configuration $\sigma$ occurring, w.r.t the mixed strategies profile $p$. The social optimum of this game is defined as $\text{OPT} = \min_{\sigma \in \Pi} \{ \max_{i \in N} \lambda^i_{\sigma_i}(\sigma) \}$. The price of anarchy for this game is then defined as $\mathcal{R} = \max_p \text{SC}(p)$ is a NE $\{ \frac{\text{SC}(p)}{\text{OPT}} \}$.

We consider atomic assignments of users to actions, i.e., each user $i \in N$ requires all its demand $w_i$ from exactly one allowable action $\sigma_i \in \Pi_i$. Nevertheless, we allow users to adopt mixed strategies. Our focus in this paper is two-fold: We are interested in families of resource delay functions for which a weighted network congestion game has a PNE, and we are also interested in the price of anarchy for a special case of this problem where $G$ has the form of an $\ell$-layered network and the delay functions are identical to the loads of the resources.

**Existence and tractability of PNE.** It is already known that the class of unweighted (atomic) congestion games (i.e., users have the same service demands) is guaranteed to have at least one PNE, since it is isomorphic to the class of exact potential games [21, 20]. In [5] it is proved that a PNE for any weighted single-commodity network congestion game (so long as the resource delay functions are non-decreasing with congestion) can be constructed in polynomial time, exploiting a nice reduction to min-cost flow. On the other hand, it is also shown that even for a symmetric congestion game or an unweighted multi-commodity network congestion game, it is PLS-complete to find a PNE (though it certainly exists). The existence and efficient construction of PNE in the special case of atomic congestion games on variants of the parallel-links network has been extensively studied in recent literature [7, 13, 18, 17, 16]. [6] initiated a different discussion concerning the problem of constructing a PNE by creating a best-response chain of configurations that ends up in a PNE, along which the social cost is at most equal to that of the initial configuration (Nashification). In [4] it is also shown that even for the unrelated parallel machines case a PNE

---

1 Since [5] only considers unit-demand users, this is also a symmetric network congestion game.
always exists, and a potential-based argument proves a convergence time (in case of integer demands) from arbitrary initial configuration to a PNE in time $O \left( mW_{\text{tot}} + 4W_{\text{tot}}/m + w_{\text{max}} \right)$ where $W_{\text{tot}} = \sum_{i \in N} w_i$ and $w_{\text{max}} = \max_{i \in N} \{ w_i \}$. [19] studies the problem of weighted parallel-edges network congestion games with user-specific costs: each allowable action of a user consists of a single resource and each user has its own private cost function for each resource.

Price of Anarchy in Congestion Games. In the seminal paper [15] the notion of coordination ratio, or price of anarchy, was introduced as a means for measuring the performance degradation due to lack of users’ coordination when sharing common resources. An extensive line of research has dealt with the price of anarchy for atomic congestion games in variants of the parallel-links network [15, 18, 7, 14, 2, 1].

In [22] the price of anarchy in a multicommodity network congestion game among infinitely many users, each of negligible demand, is studied. The social cost in this case is expressed by the total delay paid by the whole flow in the system. For linear resource delays, the price of anarchy is at most 4/3. For general, continuous, non-decreasing resource delay functions, the total delay of any Nash flow is at most equal to the total delay of an optimal flow for double flow demands. [23] proves that for this setting, it is actually the class of allowable latency functions and not the specific topology of a network that determines the price of anarchy.

Our Contribution. In this paper, we generalize the model of [15] (KP-model) to the weighted congestion games. We also define a special class of networks, the $\ell$-layered networks, which demonstrate a rather surprising behavior: their worst instance wrt the price of anarchy is (within constant factors) the parallel links network introduced in [15]. More specifically, we prove the following results:

- Weighted congestion games are not isomorphic to potential games. We show the existence of weighted single-commodity network congestion games with resource delays being either linear or 2-wise linear functions of the loads, for which there PNE cannot exist.

- There exist weighted single-commodity network congestion games which admit no exact potential function, even when the resource delays are identical to their loads.

- Any a weighted (multi-commodity) network congestion game with linear resource delays admits a PNE which can be constructed in pseudo-polynomial time.

- The price of anarchy of any weighted $\ell$-layered network congestion game with $m$ resources (edges) and resource delays equal to their loads, is at most $8 \exp \left( \frac{\log^m m}{\log \log^m m} + 1 \right)$, where $\exp$ is the basis of the natural logarithm. To our knowledge this is the first time that the price of anarchy is studied in atomic congestion games for non-trivial networks (ie, other than the parallel-links networks).

Recently we have also proved in [8] that, for the case of identical traffic demands and atomic congestion games on arbitrary single-commodity networks with edge delays proportional to the congestion, the price of anarchy is also $\Theta \left( \frac{\log m}{\log \log m} \right)$. This implies that in one more family of atomic network congestion games the parallel-links network, despite its extreme simplicity, is asymptotically the worst possible network wrt the price of anarchy.

References


On the Evolution of Selfish Routing

Simon Fischer\textsuperscript{1} and Berthold Vöcking\textsuperscript{2}

\textsuperscript{1} Universität Dortmund, Germany
simon.fischer@cs.uni-dortmund.de

\textsuperscript{2} RWTH Aachen, Germany
voecking@i1.informatik.rwth-aachen.de

Classical Game Theory is based on fully rational behavior of players and global knowledge of all details of the game under study. For routing games in large networks like the Internet, these assumptions are clearly far away from being realistic. Evolutionary Game Theory makes a different attempt to explain why large populations of agents may or may not “converge” towards equilibrium states. This theory is mainly based on the so-called replicator dynamics, a model of an evolutionary process in which agents revise their strategies from time to time based on local observations. The following paper is a highlight of DELIS-SP4 in this line of research:

We apply Evolutionary Game Theory to selfish routing in order to study the dynamics of selfish routing, rather than only the static structure of equilibria. We prove that the only existing fixed points of the replicator dynamics are Nash equilibria over a restricted strategy space. We prove that these fixed points are “evolutionary stable” which implies that the replicator dynamics converges to one of them. One standard approach in Evolutionary Game Theory to prove stability is based on symmetry properties of payoff matrices. Unfortunately, we cannot simply cast these results as our model of selfish routing allows for arbitrary latency functions whereas the existing literature on Evolutionary Game Theory assumes affine payoff functions corresponding to linear latency functions with zero offset. In fact, our proof of evolutionary stability is based on monotonicity instead of symmetry arguments.

Another novel aspect of our research is an analysis of the time it takes to reach or come close to equilibria – the time of convergence. We believe that this is an issue of particular importance as equilibria are only meaningful if they are reached in reasonable time. In fact we can prove that symmetric congestion games - corresponding to single-commodity flow - converge very quickly to an approximate equilibrium. In particular, we prove that the time for the agents to reach an approximate equilibrium is only logarithmic in the ratio between maximal and optimal latency. In addition, we present an almost matching lower bound in the same parameters. For asymmetric games, we derive upper bounds for the convergence time that are linear instead of logarithmic.

References

One of the most important concepts in non-cooperative game theory is the concept of Nash equilibria. A Nash equilibrium is a state of the system in which no player can improve its objective by unilaterally changing its strategy. A Nash equilibrium is called pure if all players choose exactly one strategy, and mixed if players choose probability distributions over strategies. The Coordination Ratio is the worst-case ratio of the Social Cost in a Nash equilibrium state and the minimum Social Cost. Of special interest to our work is the fully mixed Nash equilibrium where each player chooses each strategy with non-zero probability.

In this work, we consider a hybridization of the two most famous models for non-cooperative routing in literature: the KP model due to Koutsoupias and Papadimitriou and the W model due to Wardrop. The KP model was proposed only recently in the context of studying the effects of selfish traffic over the Internet; however, it has yet received a lot of flourishing interest and attention. In contrast, the W model dates back to the 1950s, when it was used for studying the economics of transportation networks.

The routing model considered in this work combines aspects of both, the KP model and the W model. First, we restrict to the network structure of the KP model – namely a set of parallel links – and we assume traffics to be unsplittable. On the other hand, we allow arbitrary non-decreasing and convex latency functions, whereas in the KP model latency functions are linear. In our model, the latency function of a link is a function in the total traffic of users assigned to this link. The Social Cost is defined as the expected sum of all user costs – as opposed to the Social Cost used in the KP model. Thus, as far as the generality of latency functions and the definition of Social Cost are concerned, we lean toward the W model, whereas the network structure and the indivisibility of traffics remain as in the KP model.

The convex latency functions define a very general discrete routing game. Our model is a particular instance of what is known as congestion game. It is known that a pure Nash equilibrium always exists in this setting. To the best of our knowledge this is the first time that mixed Nash equilibria are studied in such a game.

A natural question is to identify the worst-case Nash equilibrium with respect to Social Cost. We address this problem for the special case of identical users. As our main result, we prove that, in case of its existence, the fully mixed Nash equilibrium is the worst-case Nash equilibrium for any instance with convex latency functions. Therewith, we prove the Fully Mixed Nash Equilibrium Conjecture to hold for the model under consideration, whereas it remains unproven for the KP model in the general case. We use an appropriate counterexample to show that the convexity assumption we are making for the latency functions cannot be relaxed.

For the case of arbitrary non-decreasing and non-constant latency functions, we prove that the fully mixed Nash equilibrium is unique in case of its existence, and we give a complete characterization of instances for which the fully mixed Nash equilibrium exists.

For the case of identical users, we give a collection of bounds on Coordination Ratio. For pure Nash equilibria, we adapt an upper bound on the Coordination Ratio from Roughgarden and Tardos to our (discrete) model. This bound holds for non-decreasing and non-constant latency functions. For polynomial latency functions with non-negative coefficients and degree $d \in \mathbb{N}$, this yields an upper bound of $d + 1$. 
For identical links with latency function $f(x) = x^d$, the Coordination Ratio for mixed Nash equilibria is bounded by the $(d + 1)$-th Bell number. This bound can be approximated arbitrarily but never reached.

For identical users and non-decreasing latency functions, we show that a best-case pure Nash equilibrium can be computed in $O(mlgn1gm)$. It is known that for arbitrary users, computing the best-case or worst-case pure Nash equilibrium is $NP$-complete in case that Social Cost is defined as the maximum latency on a link. We show that this also holds in our model where the Social Cost is the sum of the Individual Costs even for the model of identical links with linear latency function.

References

A simple algorithm that allows selfish adaptive peers in a network to maintain high levels of cooperation whilst performing the collective task of file sharing has been developed. The algorithm is adapted from novel tag models of cooperation that do not rely on explicit reciprocity, reputation or trust mechanisms. Tags have not previously been applied to computer networks but have been used as a possible new kind of biological or cultural evolutionary mechanism that might explain certain kinds of seemingly irrational group-like behaviour in animals and humans [1].

Peer-to-Peer file sharing systems using unstructured overlay type networks have become very popular over recent years. At a given time millions of users (peers) may be running, connected over the Internet. Each peer (or node) connects to some set of known peers forming a graph type network. These applications are self-organizing in that peer software is provided freely for downloading with users deciding when to download and execute these peers on their hardware. There is no centralised administration or control and such systems are open because there is nothing stopping peers from modifying their software (and hence behavior). In addition, users can choose to share nothing or only poor quality files. Given these latter considerations and empirical evidence it can be argued that a major problem in such applications is to develop mechanisms that discourage selfish behavior, where peers download files without uploading them, and encourage altruistic behavior.

These kinds of situations, where all individuals benefit if all act altruistically but each has an incentive to act selfishly, called commons tragedies are well studied in the social sciences since they occur in many situations in naturally occurring social systems (e.g. over-grazing on a common plot of land or polluting the environment).

In P2P systems, this problem is evident in many applications, not just file sharing (e.g. the sharing of processing power or storage, the passing of messages and performing remote operations). Hence, techniques that can address the commons tragedy would appear to have wide applications within P2P systems.

In order to address this issue we adapted a recently discovered novel tag mechanism that was introduced within social simulation work as a possible theory of societal group-level altruism.

We start with an exist abstract social simulation type model and then progressively modified it to produce a simulation of a peer-to-peer type network topology performing a file-sharing task. The task allows selfish (leeching) and altruism (sharing) behaviour but the dynamic network promotes altruism - cutting off selfish nodes from the network.

From analysis of extensive computer simulations, we demonstrate the algorithm to be scalable, robust and decentralised requiring no central servers or authorities [2], [3]. The simplicity of the algorithm means peers do not need to store additional trust information about other nodes or to perform significant additional processing. Interestingly, the algorithm makes use of simple node level rules in which there is duplication and re-wiring which bear some comparison with the processes outlined previously (see Figure 1).
Figure 1: An illustration of "duplication" and "rewiring" as applied in the peer-to-peer co-operation forming algorithm. Shading of nodes represents strategy (for example a dark node may be a selfish node and a light node may be an altruistic node – helping neighbours). In (a) the arrowed link represents a comparison of utility between A and F. Assuming F has higher utility then (b) shows the state of the network after A copies F’s links and strategy and links to F. A possible result of applying rewiring to A’s links is shown in (c) and the strategy is mutated in (d). Through a "greedy" process of utility maximisation, co-operation is, counter intuitively, promoted.

References


Structure In Software Graphs

Ricard V. Solé and Sergi Valverde

1 Universitat Pompeu Fabra, Spain and Santa Fe Institute, USA
ricard.sole@upf.edu
2 Universitat Pompeu Fabra, Spain
svalverde@imim.es

Our ongoing research within SP5 is being developed towards understanding the origins of network complexity in information-processing systems and how it can be obtained under biologically-inspired approaches. In this context, we have been exploring the architecture of complex software graphs, cellular networks and in silico models of evolved circuits. All these systems appear to share some relevant features, including high degrees of heterogeneity, modular structure and some additional features that appear to come 'for free', such as degeneracy.

A first step in our approach is the exploration of the subgraph structure present in large software systems described at the class level. Since software systems are assumed to be the result of engineered design, a key assumption is that they should have been designed as efficient systems. When looking at the subgraph level, we find a set of common sub-graphs, some of them shared by all systems and others more specific. Although most of these common subgraphs are shared by some biological networks (such as genetic regulatory webs and neural maps) we have found evidence for a mechanism of subgraph generation essentially based on graph growth through duplication and rewiring (see Figure 1).

Using a simple model (previously applied to genome growth dynamics) lacking any kind of functionality – and thus not optimized for efficient performance – we have shown that it is able to reproduce the concentrations of sub-graphs found in real software architectures. Work in progress is extending this analysis towards other related systems, such as electronic circuits, both designed and evolved. By using different levels of description and considering functionality in explicit form, we hope to understand the origins of subgraph frequencies and its possible meaning, or either if they might be a by-product of network growth dynamics.

These are new results concerning software structure which may have general applicability and as such would completely transform and revolutionise many traditional assumptions of software engineering possibly leading to major advances in the understanding of the software development process.

References

Figure 1: Network motifs with \( n = 3, 4 \) elements found in software graphs. The numbers of node and edges for each network are shown. The most frequent motifs where classified in distinct rows for each type of system: medium software systems, large software systems, cellular and neural networks. For each motif, we display the number of occurrences in the network \( (N_{\text{real}}) \), the number of occurrences \( (N_{\text{rand}} \pm SD) \) in a set of 100 randomized network versions and a qualitative measure of its statistical significance (a Z score). Medium and large software networks share a large amount of motifs but we found larger variability in the medium data set. A remarkable difference is motif 2190 (the last motif in the second row), which appeared only in the context of large software systems (from [1]).

<table>
<thead>
<tr>
<th>Network</th>
<th>Nodes</th>
<th>Edges</th>
<th>( N_{\text{real}} )</th>
<th>( N_{\text{rand}} \pm SD )</th>
<th>( Z_{\text{score}} )</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Software Networks</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(medium)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FairRate</td>
<td>106</td>
<td>180</td>
<td>41</td>
<td>11.6±3.3</td>
<td>8.94</td>
</tr>
<tr>
<td>Aime</td>
<td>143</td>
<td>319</td>
<td>230</td>
<td>30.8±19</td>
<td>10.46</td>
</tr>
<tr>
<td>Flexzilla221a</td>
<td>186</td>
<td>331</td>
<td>77</td>
<td>29.5±6.1</td>
<td>7.86</td>
</tr>
<tr>
<td>Autoc</td>
<td>255</td>
<td>390</td>
<td>68</td>
<td>26.5±4.7</td>
<td>7.82</td>
</tr>
<tr>
<td>Exult</td>
<td>261</td>
<td>504</td>
<td>107</td>
<td>56.3±8.4</td>
<td>6.01</td>
</tr>
<tr>
<td><strong>Software Networks</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(large)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>blender226</td>
<td>495</td>
<td>834</td>
<td>486</td>
<td>138±30.3</td>
<td>11.4</td>
</tr>
<tr>
<td>gbk231</td>
<td>748</td>
<td>1147</td>
<td>644</td>
<td>189±33.8</td>
<td>13.4</td>
</tr>
<tr>
<td>vdk</td>
<td>771</td>
<td>1362</td>
<td>512</td>
<td>262±39.9</td>
<td>6.27</td>
</tr>
<tr>
<td>java2</td>
<td>1364</td>
<td>1947</td>
<td>816</td>
<td>189±35.5</td>
<td>17.7</td>
</tr>
<tr>
<td>prorally</td>
<td>1993</td>
<td>4987</td>
<td>22750</td>
<td>11480±171</td>
<td>122</td>
</tr>
<tr>
<td><strong>Gene Regulation</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(transcription)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E. coli</td>
<td>495</td>
<td>834</td>
<td>209</td>
<td>65±14.7</td>
<td>9.8</td>
</tr>
<tr>
<td><strong>Neurons</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Network</th>
<th>Nodes</th>
<th>Edges</th>
<th>( N_{\text{real}} )</th>
<th>( N_{\text{rand}} \pm SD )</th>
<th>( Z_{\text{score}} )</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Bi-fan</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Bi-parallel</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Bi-fan</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Bi-parallel</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| **Multi-X**            |       |       |                        |                             |                        |
|                       |       |       |                        |                             |                        |
| **Multi-X incomp**     |       |       |                        |                             |                        |
|                       |       |       |                        |                             |                        |
| **Multi-X incomp**     |       |       |                        |                             |                        |
|                       |       |       |                        |                             |                        |
| **Multi-X incomp**     |       |       |                        |                             |                        |
|                       |       |       |                        |                             |                        |
| **Multi-X incomp**     |       |       |                        |                             |                        |
|                       |       |       |                        |                             |                        |
| **Multi-X**            |       |       |                        |                             |                        |
|                       |       |       |                        |                             |                        |
| **Multi-X incomp**     |       |       |                        |                             |                        |
|                       |       |       |                        |                             |                        |
| **Multi-X incomp**     |       |       |                        |                             |                        |
|                       |       |       |                        |                             |                        |
| **Multi-X**            |       |       |                        |                             |                        |
One of the major goals of DELIS SP6 is to develop foundations for collaborative Web information search in an Internet-scale peer-to-peer (P2P) system. We are aiming at a P2P system where each peer has a full-fledged Web search engine, including a crawler and an index manager. The crawler may be thematically focused or crawl results may be postprocessed so that the local index contents reflects the corresponding user’s interest profile. With such a highly specialized and personalized “power search engine” most queries should be executed locally, but once in a while the user may not be satisfied with the local results and would then want to contact other peers. A ”good” peer to which the user’s query should be forwarded would have thematically relevant index contents, which could be measured by statistical notions of similarity between peers. Both query routing and the formation of ”statistically semantic” overlay networks could greatly benefit from collective human inputs in addition to standard statistics about terms, links, etc.: knowing the bookmarks and query logs of thousands of users would be a great resource to build on. Note that this notion of Web search includes ranked retrieval and thus is fundamentally more difficult than Gnutella-style file sharing or simple key lookups via distributed hash tables. Further note that, although query routing in P2P Web search resembles earlier work on metasearch engines and distributed information retrieval, it is much more challenging because of the large scale and the high dynamics of the envisioned P2P system with thousands or millions of computers and users.

**Top-k Query Processing**

Among the strong results achieved in DELIS SP6 so far are highly efficient algorithms for top-k queries that are suitable for a P2P environment. Top-k query processing is a fundamental cornerstone for similarity search on multimedia data, ranked retrieval on text and semi-structured documents in digital libraries and on the Web, network and stream monitoring, collaborative recommendation and preference queries, and ranking of query results on structured data sources in general. It aggregates scores for different search terms or attribute values using a monotonic aggregation function such as weighted summation, and returns the top-ranked data items as the query result. Scores are usually precomputed features of different aspects of a data item, e.g., color distributions in images, access frequencies in Web server logs, or word occurrence statistics in text documents. The state-of-the-art algorithm for top-k queries on multiple index lists, each sorted in descending order of relevance scores, is the *Threshold Algorithm, TA* for short [4, 5, 7]. It is applicable to both structured data such as product catalogs and text documents such as Web data. In the latter case, the fact that TA performs random accesses on very long, disk-resident index lists (e.g., all URLs or document ids for a frequently occurring word), with only short prefixes of the lists in memory, makes TA much less attractive, however.

In such a situation, the TA variant with sorted access only, coined NRA (no random accesses), stream-combine, or TA-sorted in the literature, is the method of choice [4]. TA-sorted works by maintaining lower bounds and upper bounds for the scores of the top-k candidates that are kept in a priority queue in memory while scanning the index lists. The algorithm can safely stop when the lower bound for the score of the rank-k result is at least as high as the highest upper bound for the scores of the candidates that are not among the current top-k. Unfortunately, albeit theoretically instance-optimal for computing a precise top-k result [4], TA-sorted tends to degrade in performance when operating on a large number of index
lists, which happens when user queries are automatically expanded based on ontologies, user profiles, or relevance feedback.

The Prob-sorted Algorithm

Statistics about the score distributions in the various index lists and some probabilistic reasoning help to overcome this efficiency problem and gain performance. In TA-sorted a top-k candidate $d$ that has already been seen in the index lists in $E(d) \subseteq [1..m]$, achieving score $s_j(d)$ in list $j$ ($0 < s_j(d) \leq 1$), and has unknown scores in the index lists $[1..m] - E(d)$, satisfies:

$$lowerb(d) = \sum_{j \in E(d)} s_j(d) \leq s(d) \leq \sum_{j \in E(d)} s_j(d) + \sum_{j \notin E(d)} high_j = upperb(d)$$

where $s(d)$ denotes the total, but not yet known, score that $d$ achieves by summing up the scores from all index lists in which $d$ occurs, $lowerb(d)$ and $upperb(d)$ are the lower and upper bounds of $d$’s score, and $high_j$ is the score that was last seen in the scan of index list $j$, upper-bounding the score that any candidate may obtain in list $j$. A candidate $d$ remains a candidate as long as $upperb(d) > lowerb(rank-k)$ where $rank-k$ is the candidate that currently has rank $k$ with regard to the candidates’ lower bounds (i.e., the worst one among the current top-k). Assuming that $d$ can achieve a score $high_j$ in all lists in which it has not yet been encountered is conservative and, almost always, overly conservative. Rather we could treat these unknown scores as random variables $S_j$ ($j \notin E(d)$), and estimate the probability that $d$’s total score can exceed $lowerb(rank-k)$. Then $d$ is discarded from the candidate list if

$$P[lowerb(d) + \sum_{j \notin E(d)} S_j > lowerb(rank-k)] < \delta$$

with some pruning threshold $\delta$. Technically, this score prediction requires computing the convolution of the score distributions in the yet to be scanned parts of the index lists. This can be implemented, for example, using histograms, fitting appropriate parametric distributions such as Poisson mixes, or using Laplace transforms of the underlying score distributions and Chernoff-Hoeffding bounds for the tail of the convolution. Figure 1 illustrates the probabilistic score predictor for early candidate pruning.

![Figure 1: Probabilistic score predictor for early candidate pruning](image)

This probabilistic interpretation makes some small, but precisely quantifiable, potential error in that it could dismiss some candidates too early. Thus, the top-k result computed this way is only approximate. However, the loss in precision and recall, relative to the exact top-k result using the same index lists, is stochastically bounded and can be set according to the application’s needs. A value of $\delta = 0.1$ seems to be acceptable in most situations. Details of this Prob-sorted method can be found in [9]. Experiments with the TREC-12 .Gov corpus and the TREC-13 Terabyte corpus (http://trec.nist.gov/), the INEX
benchmark for XML information retrieval (http://inex.is.informatik.uni-duisburg.de:2003/), and the IMDB data collection (www.imdb.com) have shown that such a probabilistic top-k method gains about a factor of ten (and sometimes more) in run-time compared to a highly tuned version of TA-sorted.

The Minerva P2P Testbed

The Prob-sorted algorithm has been implemented in the Minerva testbed for P2P Web search that is being developed within DELIS SP6 [1, 2]. Minerva uses a Chord-style distributed hash table (DHT) [8] as an overlay network. Each peer has its own search engine, along with a crawler and a local index. Peers post statistical summaries to the overlay network; they can also post bookmarks and information about their query logs and click streams at their discretion. All this information is managed as a decentralized directory, using the DHT, and is utilized by the query routing strategies. Minerva is implemented in Java and fully operational; preliminary measurements have already been performed and we plan on more comprehensive experimentation. The system will serve as a backbone for testing and evaluating algorithms developed by all DELIS partners in SP6.

Minerva can run distributed versions of both TA-sorted and the Prob-sorted algorithm. A query is collaboratively processed by a set of peers each of which holds one or more index lists for the search terms or attribute values in a query. The query-initiating peer serves as a per-query coordinator and aggregates information about top-k candidates. While such algorithms are efficient in terms of the peers’ local resource consumption, they do not pay sufficient attention to the communication costs of the computation.

The KLEE Algorithms

In joint work of CTI Patras and MPII Saarbrücken, a new family of algorithms, coined KLEE [6], has been developed to address the networking costs of top-k query algorithms. KLEE aims to minimize network latency, network bandwidth consumption, and the local work of the participating peers. To this end, it proceeds in a fixed number of phases to ensure bounded latency. In this regard, we follow the recent work of [3], but we differ significantly in the phases themselves and introduce various novel considerations.

The first phase of KLEE gathers an initial set of top-k candidates from the peers’ index lists and derives a crude threshold for the final top-k result. Along with the candidates, peers send summary information in the form of score-distribution histograms and Bloom filters for data items that locally fall into high-score histogram cells. In the second phase the coordinator performs a benefit/cost estimation for a possible additional message round that would collect further Bloom-filter information to improve the knowledge about top-k candidates. In the optional third phase this information is sent to the coordinator, which in turn prepares a refined list of candidates. In the last phase the peers are requested to send the missing scores for all data items in the candidate list above some lower bound of relevant scores. Because of the approximate nature of histograms and Bloom filters, KLEE computes an approximate top-k results, but similarly to the Prob-sorted algorithm describe above, the loss in precision and recall is small and controllable.

KLEE has been implemented in the Minerva testbed and intensively evaluated on various real-life datasets and query benchmarks. It outperforms both the distributed version of TA-sorted with batching and the TPUT method of [3] by one to two orders of magnitude when three or more peers participate in a query. This impressive performance gain is achieved by reducing both the network bandwidth consumption and the local work at the index-scanning peers. Precision and recall are 80 percent or higher, and the score and rank error measures indicate that the approximate top-k results are as good as the exact ones from a user acceptance viewpoint. This demonstrates the advantages of KLEE’s design for flexible control over different cost and query-result quality metrics, with excellent performance in terms of the quality/cost ratio.
References


Harnessing Heterogeneity in Internet-Scale Peer-to-Peer Networked Information Systems

Peter Triantafillou and Nikos Ntarmos
Computer Technology Institute, Patras and University of Patras, Greece
{peter,ntarmos}@ceid.upatras.gr

Central within the Work Packages of subproject SP6 of the DELIS Integrated Project is to investigate the development of architectures and algorithms that can provide support for exact-match and complex (including range) queries in a manner that adheres to the goals of high performance, reliability, scalability, and self-organization. The work overviewed in this document shares the same principles with related research trying to empower structured P2P data networks to deal with challenging operational characteristics typical of popular P2P applications built over unstructured networks. In this direction, we will combine research results from the Work Packages of SP6, WP6.3 and WP6.5, primarily concerned with self-organizing information dissemination and with collaborative behaviour, monitoring and accounting in P2P data networks, respectively, to showcase a highlight of the work within SP6 centered around the notion of harnessing peer capability and behaviour heterogeneity.

Our Perspectives. Our perspective is based on the fact that our world consists of altruistic people, selfish people, and of others with behaviour ranging in between, with a non-negligible percentage of the last category showing altruistic behavior if given the incentives to do so. Within the world of P2P networks this fact has been clearly manifested and documented: take as example the "free riders" [1] phenomenon, first measured in the Gnutella[6] network.

The bad news is that the great majority of peers (more than 70%[1]) are proven to be free riders. And this is indeed very bad news for DHT-style overlays, since the great majority of peers may be joining the network and leaving very soon thereafter[17, 22]. The good news are that a non-negligible percentage of the peers were proven to be altruistic[1, 12, 17]. Thus, we conjecture that, by giving incentives (to avoid the so-called tragedy of the commons) and taking away such counter-incentives, more network nodes will be willing to act altruistically.

Looking at related research in DHT-structured P2P networks, one notices that given a highly-dynamic environment, routing performance may degrades to $O(N)$ hops (that is, if the network does not become disconnected). Much to their credit, the authors in [9] studied how to guarantee in highly-dynamic cases $O(\log N)$ routing performance. To do this, $O(\log^2 N)$ so-called stabilization "rounds" need be ran by every node every half-life to update routing state (successors, predecessors, and routing/finger table entries). Furthermore, even $O(\log N)$ hops, achieved in steady-state assuming "good node behavior", may not be good enough; after all, these are overlay hops with each being translated into multiple physical network hops. In addition, even $O(\log N)$ hops over peers with low bandwidth will definitely create performance problems. Finally, within the DHT world there is a complete lack of attention on exploiting powerful peers in order to improve performance.

When considering the unstructured P2P research efforts, one also notices a lack of considerable attention on research exploiting the heterogeneities among peer nodes[17]. As an exception, [18] talk about exploiting powerful nodes, which are thought of consisting of a number of smaller, "virtual" nodes, while [10] present distributed algorithms to force-flow increased loads towards more capable nodes. But still, heterogeneity is more than a mere distinction between powerful and weak nodes; there is also heterogeneity with respect to their behavior, being altruistic or selfish.

Similar to our work, [3] are critcizing DHTs and structured overlays in failing to cope with highly-dynamic environments, such as the ones expected in sharing P2P networks. However, this led them to
avoid using structured overlays, which inevitably led to their inability to deliver definite performance guarantees, with respect to routing hop counts and robustness. Conversely, our work follows a different path; we add further structure to DHTs, with the central aim to identify and leverage altruistic/powerful peers and build an infrastructure capable of offering significant benefits with respect to performance and reliability. Our goal is twofold: (i) to create a structured architecture, at the core of which lies a subsystem, which in essence provides a ‘monitoring/accounting’ functionality, capable of efficiently and reliably identifying the altruistic peers, and (ii) to develop several additional layers, each capable of leveraging the underlying core in order to efficiently and reliably solve open problems in the related literature.

Contributions

Our work shows how to detect[13] and leverage the coexistence of altruists and selfish peers found in real-life networks and harness them to achieve high performance and reliability in fundamental system functionalities[14, 15].

Identifying Altruistic/Selfish Peers. We now outline the characteristics and distinguishing features of SeAl[13]: an infrastructure capable of identifying altruistic/selfish peers. SeAl is transparently weavable into (structured and unstructured) P2P sharing networks. SeAl components act in two ways; they provide the system with the necessary infrastructure to categorize peers and to allow them regulated access to resources, according to their contribution to the community; and they urge users to be altruistic, in order to build up a good reputation in the system.

SeAl consists of two distinct layers: (i) the SeAl monitoring/accounting layer (SAL), monitoring behavior and maintaining all metadata pertinent to the peers’ participation and contribution to the rest of the community, and (ii) the SeAl auditing/verification layer (SVL), utilizing cryptographic techniques in order to provide overwatch to the operations of the accounting layer in the presence of misbehaving users. These two layers form a substrate utilized by an incentives mechanism, which essentially increases the shared pool of content and resources.

SeAl’s monitoring/accounting functionality is based on the novel, but “natural”, notion of favors. Briefly, we say that a peer $n_1$, owes peer $n_2$ a favor $f(n_1,n_2,r)$, when $n_1$ accesses a resource $r$ shared by $n_2$ (e.g. by downloading files shared by $n_2$). Each node $n_i$, keeps a list $n_i,F_o$ of favors owed to other peers, and a list $n_i,F_d$ of favors rendered to other peers. Ideally, peers will contribute to the community the same amount of content/resources they take from it and the whole system will be in total equilibrium; thus, given similarly-sized resources, $F_o$’s and $F_d$’s must be of equal size for every peer in the network.

With this in mind, we arrive at a definition of selfishness as a function of $F_d$’s and $F_o$’s, using $|F_o|/|F_d|$ (or $|F_d| - |F_o|$). The higher (lower) the value, the more altruist (selfish) a peer is. With this infrastructure a peer $n_i$ can redirect a request for an object from a peer $n_j$ to a peer $n_k$, if $n_k$ owes a favour to $n_i$. If $n_k$ behaves selfishly and does not serve the request of $n_j$, $n_j$ posts a black record for this request for $n_k$. This is stored in an underlying DHT network, just like any other data item. In this DHT, a node is responsible for storing the black records of some other node. Further, when a request is submitted, black and white records of the requestor are submitted as well. Thus, when any peer is serving a request, it uses the white and black records of the requestor when scheduling the requests and deciding how much bandwidth to allocate when serving the request. This is in essence the core of the incentives mechanism of SeAl.

For some application environments, the above infrastructure suffices. However, other applications may be susceptible to peers willing to commit perjury, lying about their altruism score, in order to receive better service. SeAl’s auditing/verification layer is utilizing cryptographic techniques using public-private key pairs of the peers engaged in a file-retrieval transaction, to create and store sealed and verifiable records of these transactions (including black records) which can be audited at any time in order to validate any peer’s claim for an altruism score. Specifically, the serving peer randomly can choose also to verify the black and white records presented by the requestor, by retrieving the network-stored records of the requesting peer. Perjurer peers are ousted from the system, where selfish peers simply receive worse
Harnessing Altruists for Improved Routing. Once altruists have been identified, we focus on creating a hierarchical architecture, coined AESOP[14] (Altruism Endowed Self Organizing Peers) that leverages their characteristics to improve routing performance. The following figures exemplify such peer-to-peer network architectures.

As figure 1 shows, non-altruistic peers form smaller DHT clusters, with each DHT “attached” to one (or more, for reliability) altruistic peers. Altruists form a higher-level (almost) completely connected network. Altruists maintain (almost) complete routing tables. This is fine, since table entries need be very rarely updated. Using small cluster DHT sizes, (e.g., $\log N$, where $N$ is the total number of peers in the network) routing in AESOP takes place in $O(\log \log N)$. In addition, AESOP enjoys higher reliability since the highly dynamic nature of peers does not affect the whole network connectivity, just one cluster, leaving AESOP’s backbone in tact. Furthermore, stabilization within each cluster is easier, due to its smaller size.

Harnessing Altruists for Range Query Processing. Turning to efficient range query processing over DHT-based P2P networks, we have developed the RangeGuards architecture (RGA)[15]. RGA is based on our OP-Chord[20] architecture utilizing the novel notion of an order preserving hashing function to place the keys (values) of queried attributes on the underlying Chord ring. Using OP-Chord, successive values of an attribute are placed at successive nodes/peers on the Chord ring. Hence, any range query in essence defines an arc on the ring. Thus, processing any range query $(lb, ub)$ involves hashing for the value $lb$ to find and get to the starting point of the arc (costing $O(\log N)$ hops) and continue visiting successor nodes (1 hop each time) until $ub$ has been retrieved. The total hop-count cost of $O(r + \log N)$, where $r$ is the range of the query.

Figure 2 exemplifies the RGA architecture. RGs form a higher-level OP-Chord ring. Each RG node is responsible for a whole arc of the lower-level ring. Queries are sent to the appropriate RG. All successive RG peers are visited, collecting at each hop/step the values of interest that would need a much larger number of hops to be obtained at the lower OP-Chord ring.

References


A topographic view of epidemic spreading on networks

Geoffrey Canright and Kenth Engø-Monsen
Telenor R&D, Fornebu, Norway
{geoffrey.canright,kenth.engo-monsen}@telenor.com

Gossip-type protocols are of significant interest for building decentralized methods for disseminating information. Hence, as part of the effort in SP6, we have looked at some basic questions regarding spreading (of information, disease, ...) over a network. We build on earlier work [2] which develops a topographic view of a network with symmetric links. That is, we use eigenvector centrality [1] (EVC) as a 'smooth height function' over the graph. Height is then an indication of well-connectedness. The 'smoothness' of this height function then motivates the use of topographic concepts: there are mountains, defined by their peaks (local maxima of EVC); there are saddle regions bridging between the mountains; and there are low-lying edges. We call these mountains 'regions', and give them a precise definition. Our method thus analyzes any given undirected graph into a set of nonoverlapping regions which are defined completely and solely by the topology (including the possibility of weights on the links).

Since height is viewed as a measure of well-connectedness, we argue that our topographic picture is useful for understanding epidemic spreading. Our approach is thus more microscopic than typical approaches (see [5] for a review), which focus on whole-graph properties; and yet it is not completely microscopic either, as it emphasizes (using the smoothness notion) the properties of neighborhoods. We have so far tested our 'mesoscopic' model of spreading using the simplest epidemic model (appropriate also to gossip and information spreading): the 'SI' model, in which nodes are either Susceptible, or Infected. We use a constant probability $p$, per unit time, for transmission over a link from infected to uninfected node.

As a first step, we define the notion of spreading power for each node. We give a precise mathematical definition of this, which however does not allow for a general analytical evaluation. However approximations to this, plus qualitative arguments, lead us to assume that the EVC of a node is a good indicator of its spreading power. Our qualitative argument, in brief, is that a node with high EVC lies in a neighborhood that is well connected. Hence this node (and this neighborhood) should also have a high spreading power.

We also argue that spreading of an initial infection is 'uphill' – towards higher spreading power – since spreading is more rapid in regions of higher spreading power. Spreading then 'takes off' as the high part of the mountain is reached; finally it works its way back downhill, saturating the remaining uninfected areas. The result is the well known S curve of cumulative infection – but for a single region of the graph. Arguments such as these lead us to a rather detailed picture of the spreading process, which may be encapsulated in a list of predictions:

a. Each region has an S curve.

b. The number of takeoff/plateau occurrences in the cumulative curve for the whole network may be more than one; but it will not be more than the number of regions in the network.

c. For each region – assuming (which will be typical) that the initial infection is not a very central node – growth will at first be slow.

d. For each region (same assumption) initial growth will be towards higher EVC.

e. For each region, when the infection reaches the neighborhood of high centrality, growth 'takes off'.
f. An observable consequence of (e) is then that, for each region, the most central node will be infected at, or after, the takeoff – but not before.

g. For each region, the final stage of growth (saturation) will be characterized by low centrality.

Simulations. We have simulated SI epidemic spreading on a number of empirically measured social networks. We have also analyzed these networks into regions, using EVC. This enables us to test predictions a–g above.

Figure 1 shows typical spreading behavior for a snapshot of the Gnutella [4] file-sharing network, from late 2001, with about 1000 nodes. This network is very well connected, and gives a single region under our analysis. The upper curve shows the cumulative number of infected nodes over time; the circle marks the time of infection of the most central node. The lower curve of the figure shows \( \mu \) – the average EVC (height) of all newly-infected nodes at each time step. Comparing the two curves, we see that all of our predictions a–g are confirmed by the spreading behavior on this graph: the 'height' of the infection rises initially, then the infection takes off, while moving back downhill.

Figure 2 shows typical behavior for a two-region network (also taken from [4]). Cumulative and \( \mu \) curves are given for each region, along with the cumulative curve for the whole graph. We see that each region behaves as in Figure 1; but, since their takeoff times are close, the total curve for the graph is still a single smooth S curve.

Figure 3 shows spreading behavior for the Santa Fe Institute collaboration graph (taken from [3]). This graph has three regions by our analysis—and we see strong confirmation of assertion b: each region has its own takeoff and plateau, and each region’s behavior is clearly visible in the total infection curve. That is, the total infection curve (which is measured entirely independent of our analysis) clearly reveals the presence of three regions (as predicted by our analysis).
Figure 2: Spreading behavior for a graph with two regions (Gnutella)

Figure 3: Spreading behavior for a graph with three regions (SFI collaboration graph)
We believe that the most typical behavior for well connected networks will most resemble that in Figure 2: multiple regions, but strong enough inter-region connections so that the multiple S curves sum to a single S curve. This is in fact the most typical behavior seen by us in some hundreds of simulations over seven Gnutella snapshots, and three other measured social networks.

Acknowledgements. This work was partially supported by the Future & Emerging Technologies unit of the European Commission through Project BISON (IST-2001-38923), and partially supported by the EU within the 6th Framework Programme under contract 001907 (DELIS).

References


