Research Statement
Thomas S. Repantis
January 2008

Distributed systems have become part of every day life for both personal and business computing. To unleash the potential of this rapid user growth several challenges confront us. New generations of distributed systems will need to scale to unprecedented numbers of users and handle massive amounts of data. Most importantly, they will need to do so in a dependable and manageable fashion, providing real-time, highly available services. My research focuses on designing, implementing, and evaluating distributed systems that can address these challenges.

My research approach for delivering such systems lies in identifying the fundamental principles that can lead to good designs and then implementing systems that employ these principles to evaluate their performance in practice. In particular, I have focused on designing decentralized protocols that can address the scalability challenges via self-organization, and I have carried out implementations to gain complete understanding of system operation. I strongly believe that building and deploying actual systems is part of the research process, enabling the exposure of design shortcomings and the identification of performance bottlenecks.

I will first discuss my current research on Quality of Service support for distributed stream processing systems, continue by describing other research work I have completed for previous projects and internships, and conclude by exploring avenues of future work.

1 Current Research

A variety of emerging applications such as online advertising, or network traffic monitoring, require real-time processing of high-volume, high-rate data that are updated continuously. These types of applications have given rise to distributed stream processing systems. My dissertation research has focused on Quality of Service (QoS) support for such systems. Since distributed stream processing applications process data continuously and on-the-fly, adhering to QoS requirements in end-to-end delay, throughput, miss rate, or availability is crucial. Yet, providing QoS is particularly challenging, as the arrival patterns of the data to be processed are unknown, the execution environment is shared between multiple concurrently running applications, and the scale of the system makes accurate centralized decisions infeasible, since the global state is changing faster than it can be communicated to a single host. I have incorporated my techniques for QoS support for distributed stream processing systems in Synergy [17], a distributed stream processing middleware. The software prototype of Synergy is now also being used by other students as a platform for their research projects.

Sharing-aware component composition. Distributed stream processing systems need to satisfy application QoS requirements despite the shared, dynamic, and large-scale environment. I have designed and implemented in Synergy techniques for composing distributed stream processing applications with QoS demands [9]. Synergy enhances QoS provision and reduces resource load by efficiently reusing both data streams and processing components. To achieve this goal Synergy provides a set of fully distributed algorithms to discover and evaluate the reusability of available data streams and processing components when instantiating new stream applications. Probe messages travel through candidate nodes to determine whether they have enough resources available to accommodate a new application, whether the end-to-end delay achieved is within the required QoS, and whether the impact of the new application would cause QoS violations to existing applications.

For QoS provision, Synergy performs QoS impact projection to examine whether the shared processing can cause QoS violations on currently running applications. Projection is based on queueing theoretical models for both regular and bursty traffic [10]. We approximate bursty traffic with segments of data arrivals of high rate, followed by segments of data arrivals of low rate. To identify the correlation between the segments of different streams, a data arrival time series of each time stream is constructed and maintained, called the signature of the stream, that describes its workload pattern. Stream signatures enable us to combine the processing loads of multiple bursty streams.

Synergy utilizes a peer-to-peer overlay for discovering existing streams and components that are also part of a new application request, to avoid redundant computations. Using a maximum sharing discovery algorithm, the graph describing a requested application is backtracked hop-by-hop, to identify up to which point, if any, currently running applications can offer the same results.

When no existing processing components can be reused, Synergy dynamically deploys new components at strategic locations. To reduce network traffic, Synergy collocates new components with their upstream or downstream components based on selectivity. For initial component deployment Synergy employs a decentralized replica placement protocol that aims to maximize availability, while respecting resource constraints, and making performance-aware placement decisions [15].

Load prediction and hot-spot alleviation. Managing the load of the nodes of a large-scale, dynamic distributed stream processing system in real-time and without centralized supervision is challenging. Detailed monitoring of nodes' utilization, as well as frequent evaluation of complicated trade-offs are required. I have designed and implemented in Synergy a self-managing resource monitoring architecture for identifying and relieving hot-spots [13]. Monitoring responsibilities are shared among all nodes using a completely decentralized DHT-based architecture. Nodes store load data of their peers and detect overloads and load imbalances.

Nodes also proactively predict application QoS violations at run-time using a statistical forecasting framework [14]. The prediction framework binds workload forecasting with execution time forecasting. To accomplish workload forecasting, rate fluctuations are predicted by exploiting auto-correlation in the rate of each component and cross-correlation between the rates of different components of a distributed application. Linear regression is used to accomplish execution time forecasting, to accurately model the relationship of an application's execution time and of the current workload of a node.

Detecting and alleviating hot-spots in the application execution enables proactive and fine-grained load management. To alleviate both overloaded nodes and QoS-violating applications, nodes autonomously migrate the execution of stream processing components using a non-disruptive migration protocol. I have evaluated Synergy's performance over PlanetLab by implementing a network traffic monitoring application operating on real streaming data.

Managing large-scale, distributed, real-time applications. Satisfying end-to-end QoS requirements in large-scale distributed real-time applications is challenging, due to the unpredictability and heterogeneity of the environment. Focusing on media streaming and transcoding as an example, I have co-designed a decentralized architecture that enables nodes to collaborate to offer composite real-time applications [4]. Satisfying application end-to-end QoS demands is achieved via a resource management architecture that monitors current resource availability in domains [7], in cooperation with a resource allocation algorithm that distributes the processing and communication loads fairly [3, 8]. The degree of uniformity of the load distribution across nodes is captured using the Fairness Index. Using an algorithm that detects equivalent graphs representing a requested application, the graph that avoids resource overallocations and QoS violations, and maximizes the Fairness Index is selected for the instantiation of a new application.

Per-node and per-application adaptation mechanisms address changes in the infrastructure, the current resource conditions, and the user QoS requirements. Nodes adapt to variable resource conditions by optimizing their resource usage locally, selecting from discrete QoS output levels [2]. To optimize local resource usage, nodes try to maximize the output quality of the applications they are participating in under their bandwidth and processor constraints. To achieve this, they employ a utility function for each application. The overall quality of an application can fluctuate, due to varying resource availability along the different streaming paths. To combat this fluctuation, multiple nodes participating in a composite application coordinate their QoS adaptation through feedback from the service receiver.

Data dissemination in peer-to-peer systems. Peer-to-Peer systems have emerged as a cost-effective means of sharing data and services and are offering fault-tolerance and self-adaptation in large-scale environments. However, efficiently locating data or services in a fully decentralized, self-organizing, unstructured overlay network is challenging, due to the large scale and the lack of global view. I have proposed adaptive data dissemination

and content-driven routing algorithms for intelligently routing search queries in large-scale, unstructured systems [11, 16]. Nodes build and maintain Bloom filter-based synopses of their content and adaptively propagate them to the most appropriate peers. I have proposed adaptive dissemination algorithms that propagate synopses according to peers' interactions. Based on the content synopses, I have designed a routing mechanism to forward the queries to the nodes that have a high probability of providing the desired results, resulting to significant reduction in message cost.

After locating a data object or service offered by a peer, deciding whether to trust it can be challenging in the absence of a central authority. I have proposed a decentralized trust management architecture that enables peers to evaluate reputation information and avoid lying and colluding [12]. The reputation information of each peer is stored in its neighbors and piggy-backed on its replies, fully integrating the protocol with the unstructured nature of the network. For mobile peer-to-peer networks in particular, exploring the interaction of the aforementioned overlay protocols with the underlying routing protocol [1] should be an interesting part of future work.

2 Other Research

I will now describe my undergraduate diploma thesis research on page migration for software distributed shared memory systems. I will also discuss research projects I have completed while interning with IBM Research, Intel Research, and Hewlett-Packard.

Dynamic page migration for software distributed shared memory systems. Software Distributed Shared Memory (DSM) systems are an appealing alternative to message passing, since they facilitate the programmability of computer clusters. However the ease of programming comes at the expense of performance. Although accesses of data that reside to the memory of remote nodes are transparent to the programmer, they suffer from significantly higher latencies compared to local accesses. As a consequence, it is desirable to move data as close as possible to the nodes that need them most. I have proposed a protocol for dynamically migrating memory pages in software DSM systems [5, 6]. The migration protocol enables a node that heavily modifies a page to become its new home. Unlike previous work, the protocol targets multiple-writer DSMs, i.e. DSMs that allow multiple nodes to concurrently modify the same page. By implementing the protocol in a software DSM I have shown that dynamic page migration improves performance by increasing locality and adaptability, while remaining transparent to the application programmer.

Consistent replication in distributed multi-tier architectures. Multi-tier architectures are at the heart of modern data centers, offering a variety of dynamic applications. Replication is commonly employed to address the QoS requirements of multi-tier applications. While replication improves both scalability and availability, it also introduces the problem of maintaining the data consistent among the replicated servers. I have proposed an efficient distributed protocol for providing strong consistency that does not require locking or lock managers and coalesces updates. I implemented the protocol as a multi-threaded replication middleware and evaluated its performance using the TPC-W transactional web commerce benchmark under a variety of workload mixes. The experimental results for throughput and response time quantified the performance overhead of strong over weak consistency, as well as the performance benefits of replicating the data tier.

Distributed logging for asynchronous replication. Maintaining a log of updates is commonly used for achieving asynchronous replication, for resynchronization, or for failure handling in synchronous replication. Maintaining replicas consistent using a log becomes non-trivial when each replica consists of multiple nodes, each accepting requests independently. The key issue is how to take a consistent distributed snapshot without stopping user access to the data while the snapshot is being taken. I have designed and implemented a distributed logging mechanism that maintains data consistency in batches. The solution sidesteps the performance issues associated with providing a continuously consistent log by guaranteeing data consistency only at specific points.

Collaborative spam filtering robust to sybil attacks. In a sybil attack a malicious user pretends to be multiple distinct nodes in a distributed system. This enables the attacker to outvote honest users in collaborative decisions. In the case of collaborative spam filtering, by mounting a sybil attack a spammer can cause legitimate

email to be filtered as spam. I have implemented a distributed protocol that combats sybil attacks via social networks. By performing random walks on the distributed social graph, the protocol identifies suspicious subgraphs which are the result of sybil identities. I have designed and implemented an event-driven software prototype that employs this distributed protocol to offer reliable collaborative spam filtering.

3 Future Directions

I believe that a key challenge for the future will be to provide distributed systems that can keep up with the current user growth, while providing QoS guarantees. This translates to designing systems that can cope with the data volumes generated by Internet-scale applications, while providing the real-time and highly available services users are accustomed to. Thus, I plan on extending my current research that uses decentralization to address these challenges.

Efficient and consistent replication. Replication of stream processing components can alleviate performance bottlenecks or also increase the fault-tolerance of distributed stream processing applications. However, when splitting the processing load among multiple components, a fundamental trade-off exists: Consistent replication, in which components have an accurate view of each other's state at all times incurs high synchronization and communication overheads. This is particularly true for stream processing systems, that deal with high volumes of data that are updated continuously. I plan to investigate how existing stream sketching and aggregation techniques can be applied on replication and how they can be integrated with consistency protocols. Such techniques can enable accurate state reconstruction, while minimizing the amount of data that needs to be transferred between replicas. Furthermore, I plan to determine how existing consistency protocols need to be revisited to apply on such highly loaded environments, and what requirements can be relaxed with minimal effect on application accuracy, to enable efficient replication. Since a stream processing application is described by a graph, I plan on using graph theory to determine which components will benefit most the overall performance if replicated.

Adaptive topology management. As human supervision has evolved into the major factor of the total cost of IT operations, designing large-scale systems that can achieve QoS goals while being easy to manage will be of key importance. Enabling systems to make adaptive decisions regarding load and topology management can be an important step in that direction. While load management has been the focus of several research efforts, several issues remain open with regards to topology management. When overlay networks are employed, the discrepancy between overlay and physical routing, also known as overlay stretch, can affect considerably the application performance. Even when direct connections between nodes are employed, they need to be decided adaptively, according to the currently executing applications. Making these decisions at a large scale is non-trivial. For example, in a distributed stream processing system, each node hosts multiple components, each of which participates in several applications. Ideally, the connections between nodes should match the component interactions, as they evolve. Especially for stream processing applications, in which large volumes of data are transferred between nodes, topology can affect application performance in a crucial way. By running a network traffic monitoring application using Synergy over PlanetLab I have collected traces of data exchanged between different nodes of the system. I plan on making use of data mining and machine learning techniques to model component interactions, identify communication patterns, and drive adaptive topology management decisions.

Secure composite applications. Incorporating security, trust, and privacy requirements in the design of distributed systems is another important parameter to take into account as multiple entities representing businesses or individual users engage in high volumes of unsupervised transactions. Sharing data and using remote services while abiding to access control requirements poses significant challenges in loosely coupled, unstructured, and large-scale distributed systems. I plan on investigating how applications invoking multiple services hosted by different nodes can be composed while respecting data access restrictions. To achieve this, I plan on investigating how cryptographic techniques such as private information retrieval and zero-knowledge proofs can be adapted to specific application domains, and how virtualization and Service-Oriented Architecture can help in enforcing isolation and autonomy requirements.

Overlay topologies for mobile environments. Finally, an increasing number of nodes of modern distributed systems is mobile and this trend will continue to evolve. The limited resources, the heterogeneity, and the transient

connectivity of these nodes call for lightweight protocols that do not rely heavily on structure or individual nodes. An interesting question when building distributed systems for mobile environments is whether overlay topologies are robust in practice and what kind of topology best fits the physical connections. Such a study can provide design guidelines for an overlay topology that adapts to changes in the physical topology.

To conclude, I plan on continuing my research efforts by building systems that encompass design principles provided by careful analysis in the above general areas.

References

[1] I. Broustis, G. Jakllari, T. Repantis, and M. Molle. A comprehensive comparison of routing protocols for large-scale wireless MANETs. In *Proceedings of the 3rd International Workshop on Wireless Ad Hoc and Sensor Networks (IWWAN), New York, NY, USA*, June 2006.

[2] F. Chen, T. Repantis, and V. Kalogeraki. Coordinated media streaming and transcoding in peer-to-peer systems. In *Proceedings of the 19th International Parallel and Distributed Processing Symposium (IPDPS), Denver, CO, USA*, April 2005.

[3] Y. Drougas, T. Repantis, and V. Kalogeraki. Load balancing techniques for distributed stream processing applications in overlay environments. In *Proceedings of the 9th International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC), Gyeongju, Korea*, April 2006.

[4] V. Kalogeraki, F. Chen, T. Repantis, and D. Zeinalipour-Yazti. Towards self-managing QoS-enabled peer-to-peer systems. *Self-Star Properties in Complex Information Systems, Hot Topics in Computer Science, Springer LNCS*, 3460, 2005.

[5] T. Repantis, C. D. Antonopoulos, V. Kalogeraki, and T. S. Papatheodorou. Dynamic page migration in software DSM systems. In *Proceedings of the 6th IEEE International Conference on Cluster Computing (CLUSTER) (poster session), San Diego, CA, USA*, September 2004.

[6] T. Repantis, C. D. Antonopoulos, V. Kalogeraki, and T. S. Papatheodorou. A case for dynamic page migration in multiple-writer software DSM systems. In *Proceedings of the 7th IEEE International Conference on Cluster Computing (CLUSTER), Boston, MA, USA*, September 2005.

[7] T. Repantis, Y. Drougas, and V. Kalogeraki. Adaptive resource management in peer-to-peer middleware. In *Proceedings of the 13th International Workshop on Parallel and Distributed Real-Time Systems (WPDRTS), Denver, CO, USA*, April 2005.

[8] T. Repantis, Y. Drougas, and V. Kalogeraki. Adaptive component composition and load balancing for distributed stream processing applications. *Springer Peer-to-Peer Networking and Applications (PPNA)*, 2(1):60–74, March 2009.

[9] T. Repantis, X. Gu, and V. Kalogeraki. Synergy: Sharing-aware component composition for distributed stream processing systems. In *Proceedings of the 7th ACM/IFIP/USENIX International Middleware Conference (MIDDLEWARE), Melbourne, Australia*, November 2006.

[10] T. Repantis, X. Gu, and V. Kalogeraki. Qos-aware shared component composition for distributed stream processing systems. *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, 20(7):968–982, July 2009.

[11] T. Repantis and V. Kalogeraki. Data dissemination in mobile peer-to-peer networks. In *Proceedings of the 6th IEEE International Conference on Mobile Data Management (MDM), Ayia Napa, Cyprus*, May 2005.

[12] T. Repantis and V. Kalogeraki. Decentralized trust management for ad-hoc peer-to-peer networks. In *Proceedings of the 4th International Workshop on Middleware for Pervasive and Ad-Hoc Computing (MPAC), Melbourne, Australia*, November 2006.

[13] T. Repantis and V. Kalogeraki. Alleviating hot-spots in peer-to-peer stream processing environments. In *Proceedings of the 5th International Workshop on Databases, Information Systems and Peer-to-Peer Computing (DBISP2P), Vienna, Austria*, September 2007.

[14] T. Repantis and V. Kalogeraki. Hot-spot prediction and alleviation in distributed stream processing applications. In *Proceedings of the 38th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), Anchorage, AL, USA*, June 2008.

[15] T. Repantis and V. Kalogeraki. Replica placement for high availability in distributed stream processing systems. In *Proceedings of the 2nd International Conference on Distributed Event-Based Systems (DEBS), Rome, Italy*, July 2008.

[16] T. Repantis and V. Kalogeraki. *Mobile Peer-to-Peer Computing for Next Generation Distributed Environments: Advancing Conceptual and Algorithmic Applications*, chapter Data Dissemination and Query Routing in Mobile Peer-to-Peer Networks, pages 26–49. IGI Global Publishing, May 2009.

[17] The Synergy Distributed Stream Processing Middleware. `http://synergy.cs.ucr.edu`, 2007.