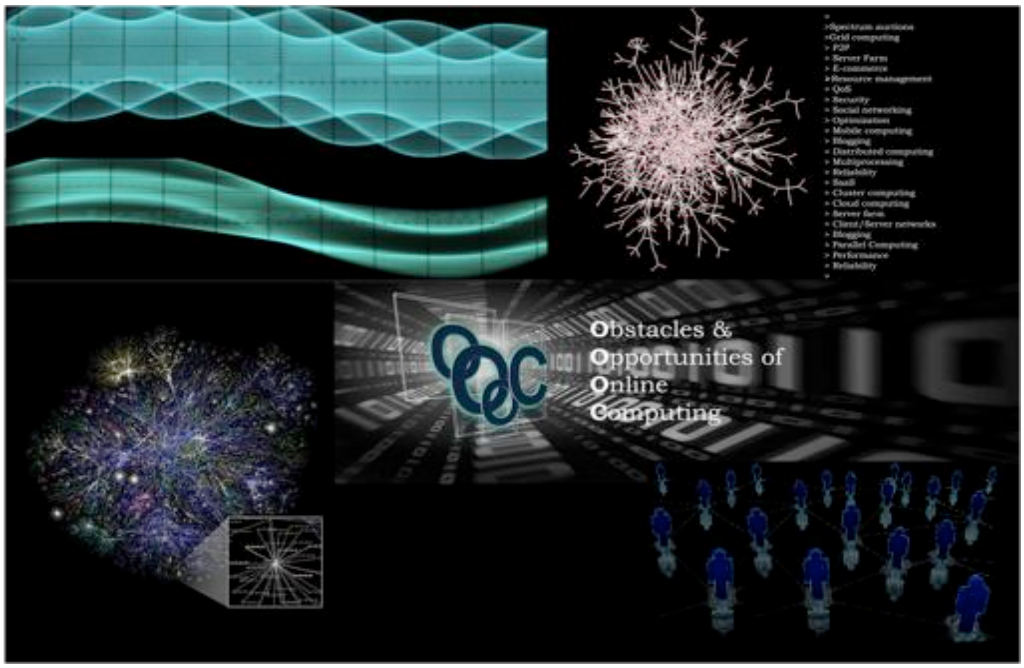
GSWC 2009



Proceedings of
The Fourth Annual Graduate Student Workshop on Computing

October 2nd, 2009
Santa Barbara, California

Thanks to:

Department of Computer Science &
Corporate Affiliates Program
University of California, Santa Barbara
http://www.cs.ucsb.edu,
http://www.industry.ucsb.edu

Organized by

Fred Chong, Advisor
Bita Mazloom, Co-Chair/Review Chair
Jonathan Ventura, Co-chair/Industry Liaison
Bryce Boe, Co-chair/Industry Liaison
Lara Deek, Publishing Chair/Industry Liaison
Christopher Coakley, General Committee/Industry Liaison
Cha Lee, General Committee
Aydin Buluc, General Committee
Hassan Wassel, General Committee
Fang Yu, General Committee

Sponsors

Keynote Speakers

Klaus E. Schauser, Appfolio Chief Strategist

Dr. Klaus Schauser is Founder and Chief Strategist of AppFolio, a new Software-as-a-Service Startup in Santa Barbara that raised $30M of VC funding. Klaus was Founder and CTO of Expertcity/CitrixOnline from 1999 through 2006. Klaus was the visionary behind GoToMyPC, GoToAssist, and GoToMeeting. Klaus led the technical teams responsible for building these products as well as the teams responsible for building and maintaining the infrastructure to securely and reliably operate the products upon. Klaus combines visionary leadership and years of expertise in overseeing technological developments. As a professor of computer science at the University of California, Santa Barbara, Klaus is a widely published research scientist with extensive experience developing scalable, highly parallel computing environments. Klaus holds a Ph.D. from the University of California, Berkeley and has received numerous academic awards.

Jan O. Pedersen, Chief Scientist for Core Search at Microsoft Bing

Jan O. Pedersen is currently Chief Scientist for Core Search at Microsoft Bing. Pedersen began his career at Xerox's Palo Alto Research Center where he managed a research program on information access technologies. In 1996 he joined Verity (recently purchased by Autonomy) to manage their advanced technology program. In 1998 Dr. Pedersen joined Infoseek/Go Network, a first generation Internet search engine, as Director of Search and Spidering. In 2002 he joined Alta Vista as Chief Scientist. Alta Vista was later acquired by Yahoo!, where Dr. Pedersen served as Chief Scientist for the Search and Advertising Technology Group. Prior to joining Microsoft, Dr. Pedersen was Chief Scientist at A9, an Amazon company. Dr. Pedersen holds a Ph.D. in Statistics from Stanford University and an AB in Statistics from Princeton University. He is credited with more than ten issued patents and has authored more than twenty refereed publications on information access topics, seven of which are in the Special Interest Group on Information Retrieval (SIGIR) proceedings.

Discussion Panel

Thorsten von Eicken, RightScale

Before joining RightScale Inc., Thorsten was Chief Architect at Expertcity.com and CitrixOnline, makers of GoToMyPC, GoToMeeting, GoToWebinar, and GoToAssist. He was responsible for the overall architecture of these online services and also managed the 24/7 datacenter operations which allowed him to acquire deep knowledge in deploying and running secure scalable online services. Thorsten received his Ph.D. from UC Berkeley and was a professor of computer science at Cornell University.

Tom Jacobs, Research Lab Manager, Adobe Systems

Tom's research interests include technologies for building Network Services (ranging from methods for exploiting cloud computing infrastructure to services for mobile devices), methods for analyzing rich-media content collections (including semantic analysis and rich media search) as well as advancing the techniques for delivering rich-media experiences (for example, next-generation content delivery systems, multi-view experiences and advanced digital rights management techniques). Prior to joining Adobe, Tom worked at Sun Microsystems (most recently at Sun Labs) on professional IPTV, broadcast and internet streaming solutions as well as managing many other network services research projects. In his early days at Sun, he led the user interface toolkit (Xview) engineering team. He also worked at Xerox where I developed parts of the Star and ViewPoint systems. I received my degree in Information & Computer Science from University of California, Irvine.

Eric Dashofy, Aerospace

Eric M. Dashofy is a Senior Member of the Technical Staff at The Aerospace Corporation in El Segundo, CA. He has a Ph.D. in Information and Computer Science from the Donald Bren School of Information and Computer Sciences, University of California, Irvine, Irvine, CA. He is one of the authors of Software Architecture: Foundations, Theory, and Practice, a full-length textbook on software architecture released in late 2008 by John Wiley and Sons. His primary research interests are in software architecture and software engineering, but he also has worked on high-performance computing, wireless sensor networks, and software patents.

Phil Lisiecki, Principal Architect for Akamai Technologies' Media Engineering Group

Over the past 10 years, Phil has contributed to almost all aspects of the Akamai technical platform. He was instrumental in laying the foundation for Akamai's distributed system and has architected and developed a variety of products for storage and media delivery. Phil is the current Chair of Akamai's Architecture Board which oversees the global architecture of Akamai's software. He recently received the Danny Lewin Award in honor of the Akamai co-founder. Phil has SB and MEng degrees in Computer Science from MIT.

David Giannini, Vice President of Engineering, Cogi

David has over twenty years of experience in the development and management of complex, reliable and scalable platforms. At Cogi, David leads a team of software engineers in the development of call recording and transcription services. Before Cogi, David was Vice President of Engineering at CallWave, where he directly developed and managed a fault tolerant service that handled millions of transactions per day (telephone and Internet) which lead to hundreds of millions in revenue. Prior to CallWave, David spent 15 years with Digital Sound Corporation as a technical lead managing the design and development of reliable and scalable platforms for voice and fax messaging solutions. Services produced by David and his teams continue to be operational world-wide. David has a BS and MS degree in Computer Science from the University of California, Santa Barbara.

Table of Contents

Table of Contents

Posters

Exploiting Locality of Interest in Online Social Networks

Mike P. Wittie, Veljko Pejovic, Lara Deek,
Kevin C. Almeroth, Elizabeth M. Belding, Ben Y. Zhao
Department of Computer Science
University of California, Santa Barbara
Santa Barbara, CA 93106
{mwittie, veljko, laradeek, almeroth, ebelding, ravenben}@cs.ucsb.edu

1. INTRODUCTION

Online social networks (OSNs), such as Facebook, MySpace, Orkut, and many others, have expanded their membership rapidly over the last several years. These networks interconnect users through *friendship* relations and allow for asynchronous communications within thus defined social graph. While various OSNs support other types of interactions, including browsing of users' *profiles*, the bulk of traffic can be attributed to inter-user communications.

OSNs continue to expand, and as a result, an ever-increasing amount of computing power and bandwidth are needed to support the communications of the growing user base. At the center of an OSN is the social graph and user data, which are traditionally stored and operated on in a centralized data center. As the result, OSN services can appear unresponsive to users located far away from such data centers.

We focus our study on Facebook, the largest OSN. Facebook's highly centralized infrastructure is not well-suited to provide services in remote areas of the globe [1]. User-generated updates are routed to the master database in California, from which they propagate to a replica in Virginia [2]. Finally, content distribution networks (CDNs), such as Akamai, serve static content to users outside of the US. While the replication of Facebook's databases provides timely updates to users in US and Europe, latency measurements from other regions confirm a sluggish service elsewhere.

In this work we aim to evaluate Facebook infrastructure design choices. Few details of Facebook's infrastructure is publicly available, and so we reverse-engineer Facebook through analysis of the history of user interactions, interaction packet traces, and network performance measurement between discovered global infrastructure endpoints. Simulations of the observed interactions within the discovered infrastructure allow us to measure a number of key performance metrics, such as transaction delays as perceived by users. We also aim to quantify server load at each point of the infrastructure and relate that information to costs of network bandwidth and data center size.

Based on the discovered shortcomings of the current design of OSN infrastructure, we propose a number of changes. While the social graph is difficult to partition, we have observed that communication patterns within the social graph are highly localized. Based on locality of traffic, we aim to show that local handling of traffic improves service responsiveness for the global user base. We also aim to show that aggregation of traffic between regional and global infrastructure can reduce load and infrastructure cost.

2. REVERSE ENGINEERING FACEBOOK

Very little information about Facebook's infrastructure is available publicly. To understand the performance of Facebook's infrastructure and the quality of service offered to its global user base, we take a three-prong approach to reverse engineer Facebook. First, we analyze user interaction history within a regional social graph. Second, we characterize Facebook's traffic through packet traces. Finally, we measure Internet path characteristics between different points of Facebook's infrastructure the iPlane system.[1]

2.1 User Interaction Analysis

Crawls of Facebook state, performed by Wilson *et al.*, show a history of user interactions and the social graph within a number of regional networks [3]. In an initial analysis of user interactions we have noticed a certain *locality of interest* within user interactions. To quantify this phenomenon and understand how it could be leveraged to improve the OSN service, we analyze user interactions by their *directionality*.

Interactions can be categorized by the locality of the interacting users with respect to a regional network. Two categories of interactions, namely local-to-local ($L \rightarrow L$) and remote-to-local ($R \rightarrow L$), can be obtained from regional network crawls by scanning each user's update *feed*. The third category of interaction involving local users, local-to-remote ($L \rightarrow R$), is calculated by multiplying the number of each user's $R \rightarrow L$ interaction by their *reciprocity factor*, or their likelihood of replying to another local user's post.

To understand the locality of Facebook's traffic, we have analyzed the directionality of interactions in three regional networks India, South Africa, and Sweden. We have selected these regions based on geographic and socioeconomic diversity and the availability of Facebook crawls.

Figure 1 shows the relative volume of $L \rightarrow L$, $L \rightarrow R$, and $R \rightarrow L$ wall posts. We observe that most of the interaction, namely $L \rightarrow L$ and $L \rightarrow R$, are generated locally, or within each regional network. The prevalence of local traffic leads us to believe that processing and caching of local traffic within each region could improve user perceived performance and reduce bandwidth load on the global Facebook infrastructure.

To understand how updates are consumed, we assume that each wall post is read by friends of both the writer and owner of the wall, which is the default Facebook behavior. We then use the social graph to calculate the number of reads, or updates Facebook needs to send out to the feeds of interested users. Figure 2 shows the ratio of read to write events for wall posts categorized by the direction of the original write. In general, locally generated traffic, or $L \rightarrow L$ and $L \rightarrow R$ interactions, exhibit a high rate of consumption within each regional network.

The analysis of regional interaction patterns shows that traffic local to a region is produced and consumed in significant

[1] iplane.cs.washington.edu

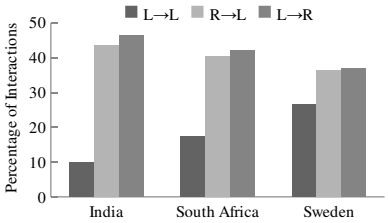


Figure 1: Wall posts.

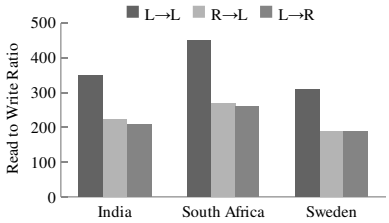


Figure 2: Wall post r/w ratio.

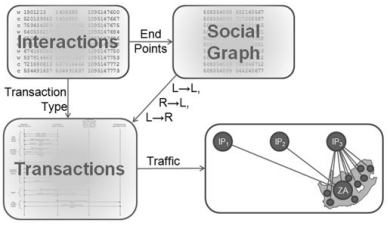


Figure 3: Simulator operation.

volumes. We believe this shows promise for a redesign of Facebook's infrastructure to cater to local interactions and, by doing so, improve user perceived responsiveness of the service and reduce infrastructure costs. To more fully understand the problem space, we continue our reverse engineering effort with the analysis of the network traffic involved in the studied interactions.

2.2 Transaction Analysis

To understand Facebook's traffic we also need to know the traffic involved in user interactions. By observing Facebook traffic of users in different regions, we can discover the addresses of servers they interact with, or infrastructure endpoints. Moreover, through sequencing and replaying of interactions we can discover caching behavior of the CDN network employed by Facebook. We capture IP packets involved and distill traffic flows between OSN server endpoints involved in each type of transaction. A transaction may contain the traffic for multiple interactions, such as the delivery of multiple wall posts, in which case we aggregate interaction data, but not transaction control traffic, within appropriate traffic flows.

2.3 Regional Network Measurement

Finally, to understand communication delays within Facebook's architecture, we measure network performance between server endpoints identified in packet traces. Packet traces are collected from a single Facebook account, but to realistically simulate connectivity between Facebook's and the many regional users, we assign these users to different subnets within each region. We identify these subnets from www.find-ip-address.org, which lists last mile subnets at a country level, and query iPlane to obtain median latency, loss, and bandwidth link characteristics between regional subnets and OSN servers.

3. EVALUATION METHODOLOGY

Our goal is to evaluate the current Facebook architecture on a network simulator driven by user interaction traces, transaction traffic, and network measurement. We would like to characterize the responsiveness of the Facebook service by measuring the delay of each transaction type. We would also like to measure server load and the amount of regionally cached data and relate these to infrastructure costs.

The high level operation of our simulator is illustrated in Figure 3. Timestamped interactions collected during crawls are precessed sequentially. Based on the interaction endpoints and the social graph, each interaction is classified as either $L \rightarrow L$, $L \rightarrow R$, or $R \rightarrow L$. Transaction type, such as 'wall post,' and its directionality allows for the identification of the corresponding traffic trace, which is then fed into a simulated network configured to represent the measured regional network. The simulator replays traffic traces of individual transactions and collects the metrics of transaction delay, and server and cache load.

The simulator will allow us to evaluate the reverse engineered Facebook infrastructure with real traffic. While this is still an ongoing effort, we propose a number of simple changes to Facebook's processing of local traffic, which we will also evaluate within our simulator framework.

4. DISTRIBUTED FACEBOOK STATE

Through analysis and simulations we have began to identify network latencies between globally distributed users and the US centralized Facebook infrastructure as the major source of user-observed delay. While Facebook mitigates some of this delay with regional CDNs, service responsiveness could be improved in two simple ways.

First, we propose that locally generated writes be handled first by a regional Facebook server before being propagated to the US for global consistency. Processing writes locally reduces transaction delay and makes the data available more quickly to other users in the regional network.

Second, we propose that communications between regional nodes and the US infrastructure be aggregated and compressed by the regional server before transmission to the US for global consistency. The aggregation of relatively contemporaneous requests will reduce network load, without introducing noticeable delays to users.

Additionally, because users now only need to communicate with the regional server, their TCP connections with US infrastructure is split by the server. Poor last-mile access links can achieve better throughput with a nearby server than with a distant server over a high-latency link and so greater interaction with regional OSN infrastructure can also improve users' effective bandwidth.

We aim to evaluate these proposed changes within the simulation framework configured through the reverse engineering effort. We hope to be able to show that the distribution of an OSN's state and traffic handling to the regional servers benefits both users and the OSN. We also hope to show that our findings generalize to other OSNs in follow up studies.

5. ACKNOWLEDGEMENTS

We would like to thanks Christo Wilson and Krishna Puttaswamy for useful discussions in the early stages of this project and for making available their crawls of Facebook.

6. REFERENCES

[1] N. Kennedy. Facebook's growing infrastructure spend. http://www.niallkennedy.com/blog/2009/03/facebook-infrastructure-financing.html, March 2009.

[2] J. Sobel. Engineering @ Facebook's notes: Scaling out. http://www.facebook.com/note.php?note_id=23844338919, August 2008.

[3] C. Wilson, B. Boe, A. Sala, K. P. N. Puttaswamy, and B. Y. Zhao. User interactions in social networks and their implications. In *EuroSys*, March 2009.

Eat All You Can in an All-You-Can-Eat Buffet:A Case for Aggressive Resource usage

Ramya Raghavendra

ramya@cs.ucsb.edu

1. INTRODUCTION

Alice walks into a restaurant with an all-you-can-eat buffet. She wants to eat enough to avoid hunger until the next meal. Should she eat based on the expected time until her next meal, or should she eat as much as she can? The second strategy is clearly superior. It provides the best possible protection against hunger, limited only by the capacity of Alices stomach. With the first strategy, misestimation of the time of the next meal or of the activity level lead to hunger. And note that both strategies cost the same.

Surprisingly, system design often follows the first strategy today. For instance, consider the task of adding forward error correction (FEC) to transmissions over a wireless channel. In current designs, the number of added FEC bits tends to be a function of the anticipated bit error rate [1, 2, 5], independent of the available spectrum resources. This method protects against packet loss as long as the errors are fewer than anticipated but fails with higher or bursty errors. This failure is unfortunate if there are available resources that would otherwise go to waste. Underlying the use of the first strategy today is a desire for efficient use of available resources. In the FEC example, adding the number of bits that is a function of the common-case error rate is an efficient way to use the spectrum. More bits might be considered wasteful usage. Yet if that spectrum would otherwise go unused, the real waste is in not taking advantage of it to improve performance. As demonstrated by the examples above, a singular focus on efficiency can lead to poor performance. Based on these observations, we put forth the Buffet principle: continue using more resources as long as the marginal cost can be driven lower than the marginal benefit. Stated differently, efficiency of resource usage should not be a driving concern if more resources can be used at a lower cost than the benefit from the additional use.

Through several case studies, we show that applying the Buffet principle produces designs that are qualitatively different and arguably perform better. Our cases span a range of systems and resource types; the diversity of these examples points to the broad applicability of the principle.

The key challenge in applying the Buffet principle is that the default way to greedily use resources tends to be costly. For example, in the FEC scenario, if the network is CSMA-based and a transmitter greedily pads its data transmissions, other transmitters will suffer and total network goodput will drop. Unless this challenge can be overcome, efficiency-oriented designs are likely prudent. We identify schemes such as prioritization, opportunistic usage when the resource is vacant, utility driven usage, and background usage as useful methods in building Buffet-based systems.

We are inspired by the end-to-end argument [4], which articulates a broadly useful principle across the design of many systems. We do not claim that the principle can be universally applied, only that it offers a useful perspective on system design. The most attractive aspect is that the performance of Buffet designs would be limited primarily by the amount of available resources, rather than likely artificial limitations driven by efficiency concerns. However, its full potential can only be understood in the context of concrete, practical designs. We are currently building two different systems based on the Buffet principle.

2. THE BUFFET PRINCIPLE

The Buffet principle is easily stated: *continue using more resources as long as the marginal cost can be driven lower than the marginal benefit*.

The simplicity of the Buffet principle is deceptive, to the extent that it might seem obvious and in wide usage. But system design today is often not approached from the perspective advocated by it. For instance, consider TCP, the dominant transport protocol for reliable communication. At first glance, it may appear that TCP uses the Buffet principle because it tries to estimate and consume all available bandwidth. However, TCP consumes all available bandwidth only if there is sufficient amount of new data, for instance, during a large file transfer. It will not use the spare bandwidth to proactively protect existing data from loss. For example, consider the case where TCPs congestion window is 8 packets and it receives only 4 packets from the application. TCP will send only 4 packets even though the path can support more, assuming that congestion window reflects available bandwidth. It will send more only after a packet is determined to be lost, which takes at least a round trip time. A Buffet-based transport protocol might preemptively send each packet twice, thus using the spare bandwidth to provide faster loss recovery. Of course, whether such a protocol is practical depends on whether other data can be protected from the aggressive bandwidth usage by duplicate packets.

As suggested by the example above, the key to successfully applying the Buffet principle is that the aggressive resource usage advocated by it must be enabled in a way that does not hurt overall performance. The default way to aggressively use resources often has a high cost; for instance, the duplicate packets above may lead to higher overall loss rate. This reason is perhaps why many system designs tend to focus on efficiency, almost by default. Approaching the design from the perspective of the Buffet principle challenges designers to devise methods to lower the impact of aggressive resource usage.

3. CASE STUDIES

We now describe several settings that can benefit from Buffet-based designs. We classify them based on the nature of the resource of interest.

3.1 Wireless spectrum or bandwidth

Forward error correction (FEC): Wireless media tends to be error-prone and the bits inferred by the receiver may be corrupted in transmission. Adding FEC bits can help recover from some of the bit errors and improve performance by reducing packet loss. The trade-off here is that each additional bit can lower packet loss but also steal transmission capacity. FEC designs that we are aware of either add a fixed

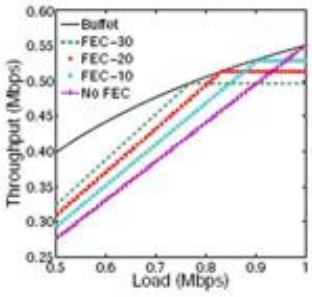


Figure 1: Throughput with different FEC mechanisms as a function of offered load.

number of bits to each transmission or a number that adapts based on estimated bit error rate (BER) [1, 2, 5]. Current designs use efficiency arguments and add bits corresponding to the sweet spot where additional bits present a diminishing reduction in loss rate. However, by not explicitly considering available resources, they either unnecessarily lose packets even when there are spare resources or create unnecessarily high FEC overhead under heavy load. Either way, throughput suffers. A Buffet-based FEC design can enable the maximal protection against bit errors that the amount of available spectrum resources can provide. Such a design will add some minimum number of FEC bits to all transmissions, perhaps based on the expected common case BER. On top of that, it will greedily add more FEC bits as long as there are spare resources.

We illustrate the benefits of such a design using a simple simulation experiment in which 8000-bit packets are sent over a channel with 1 Mbps capacity and a BER of 106. We assume an optimal FEC code: when k bits of FEC are added, the packet is successfully delivered if any 8000 out of 8000+k bits are received without error. Figure 1 shows the throughput in this setting without FEC, with different levels of added FEC, and with a Buffet design where the minimum number of FEC bits is zero. FEC-K refers to adding FEC bits equivalent to K% of the packet size current FEC designs would sit on one such curves. We see that the Buffet-based FEC performs the best across the board. For any given load level, the Buffet-based design matches the best other design. Individual other designs suffer significantly either under low load or under high load.

The example above also suggests how Buffet designs can be simpler. Current FEC designs need to carefully decide how many bits to add based on estimated BER or packet losses [1, 2, 5]. This task is complicated by the bursty and dynamic nature of the error process, and misestimations hurt throughput. Buffet designs skirt this complexity altogether by simply adding as many bits as the currently available resources allow, they can get the best performance at all load and BER levels.

Rationale similar to the one above also applies to protection against packet losses. A Buffet-based system can provide greater protection from losses by utilizing all remaining path capacity for erasure coded packets.

Mobility updates: The performance of systems that exhibit a high-degree of mobility, such as a mobile ad hoc network (MANET), depends on the frequency of mobility updates. These solutions suffer from poor performance in high mobility conditions, even when spare resources are available. A Buffet-based mobility update mechanism will provide better performance whenever spare capacity is available.

Routing in delay tolerant networks (DTNs): DTN routing protocols replicate messages along multiple paths to improve their delivery probability. Because this limit is not guided by the amount of available storage or bandwidth between replication end points, these designs can perform poorly even when plenty of resources are available. A recent protocol, called RAPID [3], implicitly uses the Buffet principle. It replicates as much as available resources allow and as a result outperforms the conventional appraoches significantly.

3.2 Storage

Long-term storage: The amount of replication on disk, is often pre-determined today, based on anticipated failure rate. This unnecessarily limits the protection level even when there may be spare resources. A replication system based on the Buffet principle will provide maximal protection given available resources by aggressively replicating to fill all available storage.

Short-term storage: Program execution can be slowed by the time it takes to load the program and the libraries it uses into memory. A Buffet-based strategy will maximize performance by aggressively filling available memory, instead of being limited to the most promising candidates.

Computational resources: Speculative execution is a commonly used technique wherein parts of code are executed even though the results may eventually be discarded. A Buffet design would speculatively follow as many paths as current resources levels allow. As the number of cores inside processors increase, such a design would increasingly outperform strategies that limit speculative execution to more likely paths.

4. CURRENT WORK

Currently, we are in the process of providing wireless protocol designers with a new primitive, the Buffet primitive. Using the Buffet primitive, we are designing a new 802.11-compatible MAC that allows a sender to use all *available* capacity on a specific channel, without hurting other users on the channel. The motivation behind the BuffetMAC system is the Buffet principle: the goal is to consume all the *available* air time on the channel. BuffetMAC builds upon the basic priority mechanism offered by the IEEE 802.11e standard. The 802.11e standard offers four classes of priorioties. The prioritization is enabled primarily by using higher AIFS window for low priority packets. However, the priority mechanism offered by the 802.11e MAC is not sufficient to build BuffetMAC.

5. REFERENCES

[1] J.-S. Ahn, S.-W. Hong, and J. Heidemann. An Adaptive FEC Code Control Algorithm for Mobile Wireless Sensor Networks. *Journal of Communications and Networks*, 7(4), 2005.

[2] A.Levisianou, C.Assimakopoulos, N-F.Pavlidou, and A.Polydoros. A Recursive IR Protocol for Multi-carrier Communications. *International OFDM Workshop*, September 2001.

[3] A. Balasubramanian, B. Levine, and A. Venkataramani. DTN Routing as a Resource Allocation Problem. August 2007.

[4] J. H. Saltzer, D. P. Reed, and D. D. Clark. End-to-end Arguments in System Design. *ACM ToCS*, 1984.

[5] L. Zhao, J. W. Mark, and Y. Yoon. A combined link adaptation and incremental redundancy protocol forenhanced data transmission. *GLOBECOM*, November 2001.

Adaptive Binning of Oriented Histograms of Gradients

Justin Muncaster
UCSB Four Eyes Lab
jmunk@cs.ucsb.edu

Matthew Turk
UCSB Dept. of Computer Science
mturk@cs.ucsb.edu

1. INTRODUCTION

As algorithms for tracking targets at close ranges in video have matured, there as been increased interest in the classification of objects in video. Algorithms for objects in high resolution imagery such as faces, pedestrians , and vehicles have all come a long way in recent years ([5, 2, 4]), however most techniques are geared towards applications in which there are a large number of pixels on the object of interest and the imagery is of relatively high quality.

In surveillance applications it is very common to have both poor image quality and low resolution. These conditions stem from the fact that the goal in surveillance is often to track individuals at the furthest range possible. The ability to track and classify individuals at increased ranges allows for larger areas to be observed, larger amounts of time to be spent observing a given target, and increased time to respond to events. It allows unmanned air vehicles (UAVs) to fly higher and with increased safety, traffic monitors to gather more data per vehicle, and border patrol agents to locate potentially threatening activity with enough time to intercept it.

Additionally, imaging at increased ranges necessitates imaging in outdoor environments where conditions are often far from ideal. Lighting changes, shadows, wind, haze, thermal distortion, dust, and occlusions all become more apparent as targets are imaged at increased ranges.

Although camera resolutions and camera optics may improve over time, there will be a continual demand to observe individuals at increased ranges, effectively cancelling out many of the benefits of increased resolution. Even as camera optics improve, the environmental factors that degrade image quality will only become more apparent as operators try to apply algorithms on targets at increasing distances.

In this work, we look at features for classification at far ranges. We begin with a preliminary examination of oriented Histogram of Gradient (HoG) features, which are robust to scale change and small rotations. We use these features in the popular Adaptive Boosting framework and show that we can improve classification performance at a fixed binning resolution by translating the bin locations. Next, we examine a novel feature type which we call a Cumulative Histogram of Gradients Interval and examine the performance gained by allowing the classifier to automatically choose feature resolution.

2. SENSIAC DATASET

To do our experiments we use the SENSIAC dataset ([1]). This dataset contains video captured using a long-wave infrared (LWIR) camera and a black and white camera capturing the visible spectrum. The LWIR video was taken both during the day and at night, whereas the visible spectrum video was taken only during the day. In an effort to develop a classifier which works well in many situations, we use data from both camera sources and at both times of day in this work. The video is obtained at 500 meters, 1000 meters, and 1500 meters and contains humans and vehicles. Table1 describes the type of video present.

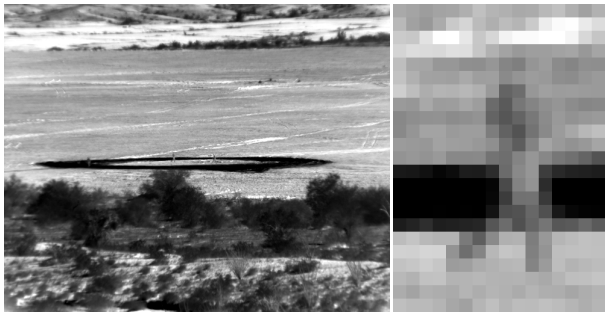


Figure 1: Example frames of humans at 1000 meters in LWIR video during the daytime. To the right is a blown up image of a human in the frame. The human is 16x36 pixels.

Table 1: Videos used from the SENSIAC dataset.

Video Contents	Ranges
Two humans walking in figure-eight	500m, 1000m
Two humans jogging in figure-eight	500m, 1000m
Pickup truck driving in circle	1000m, 1500m
SUV driving in circle	1000m, 1500m

In Figure 1 we show an example frame from the data set and a blown up image of the human in the video. This frame is relatively clean, which makes it good for studying the effects of poor resolution independently from the effects of image degradation. Notice the low resolution on the image of the human.

3. HISTOGRAM OF GRADIENT FEATURES

In order to do classification we will extract features from pixels located within the ground-truthbounding box of each object. The features we extract will be oriented histograms of gradients. To extract these features we first compute for each image chip the gradient image in the x-direction and y- direction. Next, we compute the magnitudes and directions of the gradients. Finally, we consider only the pixels whose gradient magnitude is above the median gradient magnitude.

Next, we consider a histogram over the possible gradient directions, where the gradient direction at each pixel is given by $\theta_{i,j} = atan2(\delta_y, \delta_x)$, where δ_x denotes the image intensity gradient in the x-direction at image location (i, j). We then normalize $\theta_{i,j}$ to be between 0 and π so that light-dark edges and dark-light edges are treated uniformly. Next, we choose a number of bins N_{bins} for our histogram and tally up how many

gradients fall within each bin. Finally, to normalize for rotation we rotationally-shift the histogram so that the maximum bin is located at the first bin.

We also introduce a new novel feature which is based on the cumulative probability mass function obtained from the HoG feature. We integrate the histogram bin values over a continuous interval $[t_0, t_1]$, and consider intervals of increasing size. To compute this feature we start with an oriented histogram $h_t,\ t = 1 \ldots N_{bins}$, and compute:

$$c_{t,w} = \sum_{k=t}^{t+w} h_{k'}$$

where $k' = 1 + mod(k, N_{bins})$ is the histogram bin which results from wrapping the indexing variable around to the early indices. We enumerate all values of $t \in \{1 \ldots N_{bins}\}$ and $w \in \left\{1 \ldots \frac{N_{bins}}{2}\right\}$ and use these as candidate features for classification. This procedure essentially allows a feature-selection algorithm such as AdaBoost to perform adaptive binning and multi-scale analysis when choosing histogram features, as it will choose features with large and small values for w to classify using coarse and fine information.

4. PRELIMINARY RESULTS

Place We use the AdaBoost maximum-margin machine-learning algorithm [3] to train a single classifier which is a weighted sum weak perceptron classifiers. Each perceptron classifier classifies on the basis of a single feature. Thus, this classifier's performance is directly related to the quality of the features. We trained each classifier on 12 clips (6 humans, 6 vehicles) and tested on a separate set of 12 clips (6 humans, 6 vehicles). We made sure to include all varieties of videos in both the training and testing set (LWIR, visible spectrum, daytime, nighttime, humans, pickups, suv).

We show classification results using receiver operating characteristic curves (ROC-curves). These curves show the tradeoff between false alarms and missed detections achievable by varying the decision threshold. Better results stretch the curve closer to the upper-left corner of the graph. In Figure 2 we show results achieved when keep the bin-width constant and limit the number of features to be chosen to eight. We obtain superior performance than the plain HoG feature by simply allowing for the bin locations to be optimized using AdaBoost rather than fixed *a priori*. In Figure 3 we show ROC curves comparing the use of our adaptive binning feature mentioned in Section 3. Our feature allows the AdaBoost feature-selection algorithm to control for overly-sparse histograms and to essentially choose a resolution coarseness which yields superior classification results.

5. CONCLUSION

In this paper we showed a how the classification performance of a classifier based upon the popular histogram-of-gradients can vary with the bin granularity. We showed that in low-resolution imagery, coarse binning is preferable to fine-grained binning. We also presented a new feature type which is based on a cumulative sum over the oriented histogram of gradients, and preliminary results show that it improves classification performance. In future work we intend to examine this feature and how it compares to other state-of-the-art features for classification. We also plan to more thoroughly examine methods for dealing with low-resolution and poor-quality imagery.

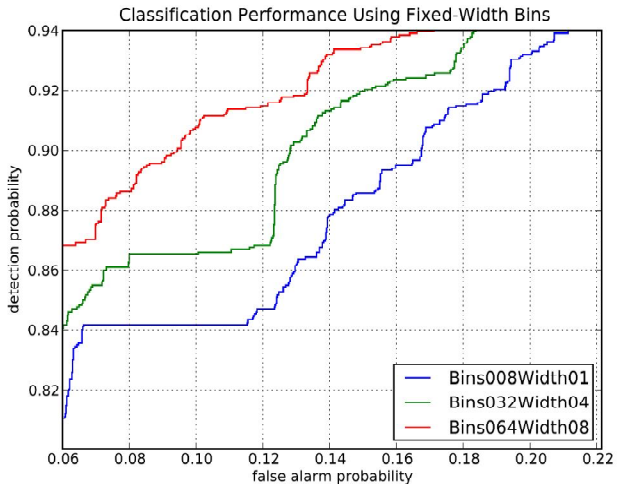


Figure 2: ROC curve shows classification results using cumulative histogram of gradient features with a fixed bin width. False alarm probability is on the x-axis and detection probability is on the y-axis.

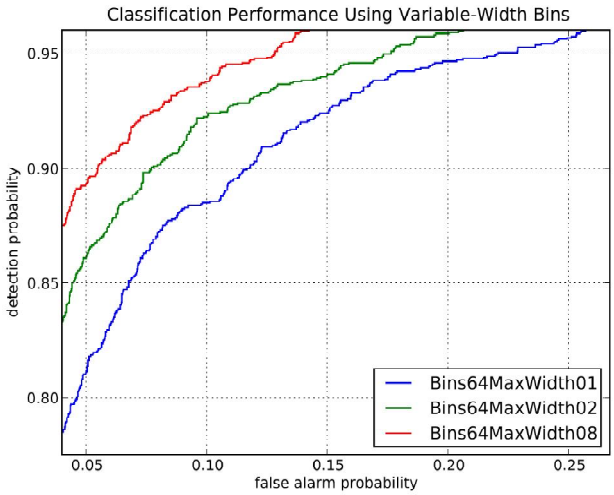


Figure 3: The ROC curve for our cumulative histogram of gradients. Notice that the increased use of the adaptive binning aspect of our feature increased the classification performance.

6. REFERENCES

[1] ATR Algorithm Development Image Database. https://www.sensiac.org/external/index.jsf.

[2] C. Papageorgiou, M. Oren, and T. Poggio. A general framework for object detection. Computer Vision, 1998. Sixth International Conference on, 1998.

[3] R. Schapire, Y. Freund, P. Bartlett, and W. Lee. Boosting the margin: A new explanation for the effectiveness of voting methods. ANNALS OF STATISTICS, 1998.

[4] P. Viola and M. Jones. Robust real-time object detection. International Journal of Computer Vision, 2002.

[5] P. Viola, M. Jones, and D. Snow. Detecting Pedestrians Using Patterns of Motion and Appearance. International Journal of Computer Vision, 2005.

Predicting Network Flow Behavior From Five Packets

Stefan Karpinski, John R. Gilbert, Elizabeth M. Belding

Department of Computer Science
University of California, Santa Barbara

{sgk,gilbert,ebelding}@cs.ucsb.edu

ABSTRACT

We observe that when network traffic behaviors are represented in vector spaces as relative frequency histograms of behavioral features, they exhibit low-rank linear structure. We hypothesize that this structure is due to the distribution of flow behaviors following a finite mixture model. Aside from being of theoretical interest, this hypothesis has practical consequences: it allows us to make predictions about the probabilities of future flow behaviors from a handful of a flow's initial packets. From observing five initial packets, we are able to predict the distribution of future packet sizes and inter-packet intervals with between 70% and 90% accuracy across a variety of network traces. We can predict which flow will have more packets in pairwise comparisons with between 65% and 85% accuracy. These practical applications serve dual functions. They provide highly useful tools for network management, routing decisions, and quality of service schemes. However, they also provide evidence that the hypothesized model gives a correct explanation for the observed linear structure in real network traffic.

Extended Abstract

This work begins with a particular way of representing flow behaviors as vectors. The representation is quite simple. For each feature of a flow, we represent that aspect of the flow's behavior as a *feature-frequency vector*: a vector having a dimension for each possible value of the feature and whose coordinates are the relative frequency of values. For example, the vector for the distribution of packet sizes of a flow with four 40-byte and two 145-bytes packets is

$$\text{size} = \frac{1}{4+2}(4\mathbf{e}_{40} + 2\mathbf{e}_{145}). \tag{1}$$

Different aspects of flow behavior can be represented in this way, and these representations can be combined by taking the direct sum of their representation vectors:

$$\text{flow} = \text{size} \oplus \text{ival} \oplus \text{type} \oplus \text{port} \oplus \text{pkts}. \tag{2}$$

The features here are packet size and inter-packet interval distributions, IP protocol type, source and destination port numbers, and packet count. The behavior of a feature across a collection of flows can be expressed as a matrix where each row represents a flow:

$$\text{Size} = [\ \text{size}_1\ ;\ \cdots\ ;\ \text{size}_m\ ]. \tag{3}$$

The overall behavior of the collection of flows then becomes a concatenation of these feature matrices:

$$X = [\ \text{Size Ival Type Port Pkts}\ ]. \tag{4}$$

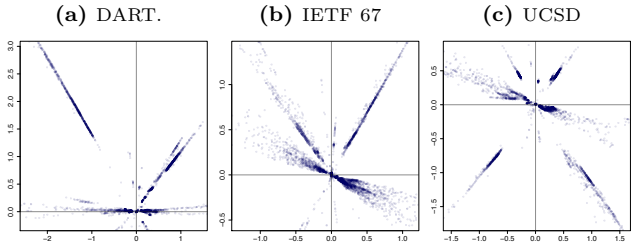


Figure 1: Scatter plots of the two most significant SVD dimensions of the feature-frequency representations of traffic samples from the six network traffic traces analyzed in our experiments.

When traffic traces are represented like this, a very curious thing happens: the resulting matrices exhibit a great deal of linear structure. Specifically, flow behaviors tend to lie near the union of a small set of low-rank subspaces. Figure 1 shows this structure visually. These scatter plots show the first two most significant dimensions of the behavior matrix after reduction via singular value decomposition (SVD) and projection onto the unit-sum hyperplane.

To explain this linear structure, we hypothesize that the behavior distribution for most flows is a mixture of a small set of "basic behaviors." Moreover, only even smaller subsets of these basic behaviors are typically combined with each other. Under these assumptions, we can express the distribution of each flow's behaviors as a finite mixture model [1]:

$$q_i(x) = \sum_{j=1}^{r} w_{ij} p_j(x). \tag{5}$$

Here q_i and p_j are probability density functions, and w_{ij} are nonnegative weights, summing to unity for each i. Equation 5 is expressed succinctly as matrix multiplication. Writing $Q_{ik} = q_i(k)$, $W_{ij} = w_{ij}$, and $P_{jk} = p_j(k)$, we have:

$$Q = WP. \tag{6}$$

The number of basic behaviors, r, is the maximum possible rank of the feature distribution matrix, Q. Moreover, we can partition the rows of P into classes such that w_{ij_1} and w_{ij_2} are both non-zero only if j_1 and j_2 are in the same class. Thus, each row of Q is associated with exactly one class, and all the points associated with a class lie in the subspace spanned by its associated rows in P.

This model explains the structures in Figure 1. Points along the same low-rank structure are in the same class. A structure is "generated" by a small set of vertices: points belonging to a structure are near the hull of its vertices. This is only one possible hypothesis that fits the data. Like any hypothesis, it must be tested. Our prediction technique, aside

Trace	Year	Type	Network
DARTMOUTH	2003	campus	Dartmouth College
IETF 60	2004	conference	IETF hotel
IETF 67	2006	conference	IETF hotel
SIGCOMM 2001	2001	conference	SIGCOMM hotel
SIGCOMM 2004	2004	conference	SIGCOMM hotel
UCSD	2007	campus	UCSD engineering

Table 1: Traffic traces used for analysis and experiments.

from providing a practical application, serves as a hypothesis test: we try to recover the matrices W and P from our noisy and imperfect observations of Q and use the recovered model to predict real flow behaviors. If the recovered model can make accurate predictions, this provides evidence that our model and hypothesis approximate reality.

From training data we recover estimates, W^* and P^*, of the factors in Equation 6. To detect the low-rank linear structures, we use Ma *et al.*'s algorithm for segmenting multivariate data into subspaces using lossy data coding and compression [2]. Then we determine the hull points of each linear structure using nonnegative matrix factorization (NMF) [3, 4]: if Q_c is a sub-matrix of rows in the same structure class, we want to find nonnegative matrices, W_c and P_c, such that $Q_c \approx W_c P_c$. Our reconstructed P^* is a vertical concatenation of these P_c matrices, while W^* is a row-permutation of the direct sum of W_c matrices. We use Kim and Park's alternating non-negative least squares algorithm [4] for rapid initial convergence, but refine the result using Lee and Seung's Euclidean algorithm [3]. Good prediction performance requires special initialization of the NMF algorithms, using new techniques that we lack room to detail here.

To predict flow behavior, we separate flow features into those observed and those to be predicted:

$$X_\text{o} = [\ \text{Size}_\text{init}\ \text{Ival}_\text{init}\ \text{Type}\ \text{Port}\ ], \quad (7)$$

$$X_\text{p} = [\ \text{Size}_\text{rest}\ \text{Ival}_\text{rest}\ \text{Pkts}\ ]. \quad (8)$$

Size_init is the packet size matrix for the first five packets, while Size_rest is the matrix for the remainder of the packets, and similarly for inter-packet intervals. From an observation matrix, X_o, and the recovered model parameters, P^*, we can make predictions about X_p. Let $P^* = [P^*_\text{o}\ P^*_\text{p}]$ be the recovered model parameter matrix with separated observable and predictable features. From an observation matrix for test data, X_o, we estimate the matrix of weights by minimizing the squared Frobenius error:

$$W^* = \text{argmin}_W \|X_\text{o} - W P^*_\text{o}\|^2_\text{frob} \quad (9)$$

with the constraint that W be nonnegative. We can estimate the underlying feature distributions for the flows:

$$Q^* = W^* P^*. \quad (10)$$

The "predictable" portion, Q^*_p, contains predictions of packet size distribution, inter-packet interval distribution and distribution of packet counts for each flow. To evaluate the quality of these predictions, we compare the distributions in Q^*_p to the matrix, X_p, of actual test flow behaviors.

For our experiments, we use randomly sampled traffic from six network traces. The traces are freely available from the CRAWDAD trace repository [5]. Details of the traces are shown in Table 1. We randomly sampled 5000 flows from each trace for training and another 5000 flows for testing. The results are shown in Figure 2. The left panel shows accuracy rates for predicting packet size and inter-packet interval distributions. Since no previous work attempts to either model individual flow behavior or predict flow behavior from initial observations, we compare our prediction technique to two simple and obvious approaches: predicting the already observed behavior of each flow, and predicting the average behavior of the training trace. Accuracy is computed by comparing the distribution of Kolmogorov-Smirnov (K-S) test p-values to an empirical ideal distribution of K-S p-values, taking the maximum deviation from the ideal as the error rate. The right-top panel shows the accuracy rate for predicting which flow will have more packets between random pairs of flows. Choosing randomly gives 50% accuracy, which we exceed significantly on all traces.

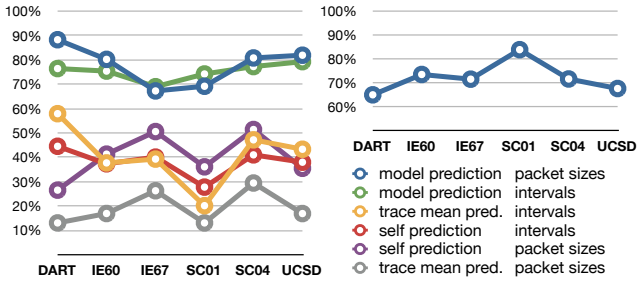


Figure 2: Accuracy rates of various methods of predicting flow behavior from five initial packets across six data sets.

Our method does not yield perfect predictions, but the nondeterministic nature of flow behavior implies that it is impossible to achieve perfect prediction. Moreover, we do not know what the inherent upper limit on prediction quality is. No prior work has provided detailed statistical models of individual flow behaviors, or attempted to predict individual flow behavior from initial packets. The fact that this technique can accurately predict flow behavior from so few initial packet observations is evidence that our hypothesized mixture model for flow behavior has merit. With improvements in the algorithms used to recover the model parameters, we are confident that even better prediction accuracy can be achieved. Furthermore, the same model can be applied to traffic classification from flow behavior, and to generation of realistic synthetic network traffic from collections of trace data. These applications, however useful, are merely pleasant side effects of the real breakthrough of this work: a realistic, detailed statistical model for individual flow behaviors across whole networks.

REFERENCES

[1] G. McLachlan and D. Peel. *Finite Mixture Models*. John Wiley and Sons, New York NY, USA, 2000.

[2] Y. Ma, H. Derksen, W. Hong, and J. Wright. Segmentation of multivariate mixed data via lossy data coding and compression. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(9), September 2007.

[3] D. Lee and H. Seung. Algorithms for non-negative matrix factorization. *Advances in Neural Information Processing*, 13, 2001.

[4] H. Kim and H. Park. Non-negative matrix factorization based on alternating non-negativity constrained least squares and active set method. *SIAM Journal in Matrix Analysis and Applications*, 30(2), May 2008.

[5] J. Yeo, D. Kotz, and T. Henderson. CRAWDAD: a community resource for archiving wireless data at Dartmouth. *SIGCOMM Computer Communication Review*, 36(2), April 2006.

Online Environment Model Estimation for Augmented Reality

Jonathan Ventura
Four Eyes Lab

1. INTRODUCTION

Augmented reality applications often rely on a detailed environment model to support features such as annotation and occlusion. Usually, such a model is constructed offline, which restricts the generality and mobility of the AR experience. In online SLAM approaches, the fidelity of the model stays at the level of landmark feature maps. In this work we introduce a system which constructs a textured geometric model of the user's environment as it is being explored. First, 3D feature tracks are organized into roughly planar surfaces. Then, image patches in keyframes are assigned to the planes in the scene using stereo analysis. The system runs as a background process and continually updates and improves the model over time. This environment model can then be rendered into new frames to aid in several common but difficult AR tasks such as accurate real-virtual occlusion and annotation placement.

2. MODELING

2.1 Finding planes

To find planes in the environment, we look for planar groups of well-estimated feature points. A SLAM tracking system is one source of such points, especially indoors [4]. A laser range finder would be more appropriate for outdoor AR, where points need to be sampled from a greater distance than multi-view matching can handle [6].

Once we have assembled a set of 3D estimated points from the environment, we find groups of points which roughly lie on planes. We allow some error in the planar fit, to allow for error in the point estimates and also for surfaces that are not perfectly planar. The RANSAC estimation procedure is well-suited to robustly finding planes in the point set [3].

When we add a new point to the map, we first see if the point can be classified as an inlier to a plane. If the point is not an inlier to any known plane, the point remains unlabeled. During our plane re-estimation procedure, we try to find new planes in the set of unlabeled points, using RANSAC as described above. Then all planes are re-estimated based on their sets of inliers.

2.2 Texturing planes

Once the planes have been estimated, we need to determine to what extent each plane is visible in the keyframe images. For each particular keyframe, we will assign one plane to each pixel.

It is well-known that given a plane, we can project one image observation onto another using a linear homography [5]. We test plane hypotheses by projecting a matching frame into the reference using the corresponding homography (with radial distortion applied). For each projection of a keyframe, we calculate the per-pixel matching error by Euclidean distance in RGB space.

Several recent multi-view modeling papers have shown the advantages of using over-segmentation (superpixels) for image matching [1, 7]. The over-segmentation naturally respects image edges, and aggregating over a larger region can help with matching texture-less regions. We use a recent image segmentation method which can segment a 640×480 image in less than one second on a 2.1 GHz machine [2]. Real-time speeds are not necessary since the reconstruction runs as a background thread.

After segmenting the reference image, we can aggregate matching costs from the image warping described above. We sum up per-pixel matching costs over each segment, and for each segment choose the plane resulting in the lowest cost. After segmentation, multi-view matching and patch label assignment takes under 0.5 ms per plane.

3. VIEW SYNTHESIS

After plane detection and multi-view matching, each keyframe contains its own planar model of a portion of the environment. Given any camera pose, we can synthesize a new view of the environment from one or more keyframes.

Our rendering algorithm combines nearby keyframes together to produce a synthesized view. We may need to use more than one keyframe to cover the entire extent of the desired camera view. Up to four of the nearest keyframes (as determined by the ranking described in Section 2.2) are rendered from the viewpoint to be synthesized. The rendering procedure is similar to the image matching procedure described in Section 2.2. For each plane and each nearby keyframe, we calculate the homography H which projects the matching frame into the reference frame. However, only pixels which are assigned to that particular plane are projected (with radial distortion). At a single pixel location, we collect all the projected pixels from nearby frames. We output the pixel with minimum color matching error, calculated by Euclidean distance in RGB space.

3.1 Dynamic real-virtual occlusion

One useful benefit of our online modeling approach is its ability to detect when foreground objects such as hands occlude the background, and thus should occlude rendered virtual objects as well. Figure 1 shows this effect.

We calculate the occlusion map by thresholding the differences between the camera image and the synthesized view. The differences are already calculated and used by the rendering algorithm described in Section 3, so our occlusion approach adds little overhead.

3.2 Annotation with the environment model

A second benefit of our modeling and rendering system is that we can estimate both depth and a normal direction for each pixel of the camera image, in real-time. This has the potential for greatly improving AR interaction with unknown environments, without any offline modeling of the environment.

Our view synthesis method assigns to each pixel in the camera image a plane from the set of detected planar surfaces. Using ray casting, we can calculate the depth of a pixel by intersecting with its plane in world space, after accounting for radial distortion. The normal is given by the plane normal.

With this technique, the user can easily add annotations to the environment by simple point-and-click, which is an important AR task. In Figure 1 we show how a text banner can be placed on any of the planes detected by the system.

4. CONCLUSIONS AND FUTURE WORK

We have introduced a novel system for online modeling of arbitrary AR environments, by multi-view analysis of detected planar surfaces. We demonstrated our system's effectiveness for improving the augmented reality experience. Our approach is fast and can be used as part of a live system. The model updates and improves as more frames are added, and can expand as the user explores the environment. With our image-based rendering method, our model can be used for simplified annotation which is automatically oriented to the model, and foreground object occlusion detection.

Our multi-view matching and rendering methods are robust, but still susceptible to failure due to either significant illumination changes, or physical changes to the environment. It would be interesting to investigate how we can relax the requirement of a static environment, and enable the system to better adapt to a dynamic scene. For example, we could remove old data from the model, or choose between different versions of the model depending on what is sensed by the camera.

Figure 1: Annotation and occlusion. *Top:* Based on the difference between the camera image and the synthesized view, we detect foreground objects which occlude virtual objects. *Bottom:* Using plane detection, we add an oriented annotation to the environment.

5. REFERENCES

[1] M. Bleyer and M. Gelautz. Graph-cut-based stereo matching using image segmentation with symmetrical treatment of occlusions. *Image Commun.*, 22(2):127–143, 2007.

[2] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient graph-based image segmentation. *Int. J. Comput. Vision*, 59(2):167–181, 2004.

[3] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, 1981.

[4] G. Klein and D. Murray. Parallel tracking and mapping for small ar workspaces. In *ISMAR '07: Proceedings of the 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, pages 1–10, Washington, DC, USA, 2007. IEEE Computer Society.

[5] Q.-T. Luong and T. Viéville. Canonic representations for the geometries of multiple projective views. In *ECCV '94: Proceedings of the third European conference on Computer vision (vol. 1)*, pages 589–599, Secaucus, NJ, USA, 1994. Springer-Verlag New York, Inc.

[6] J. Wither, C. Coffin, J. Ventura, and T. Hollerer. Fast annotation and modeling with a single-point laser range finder. In *International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 65–68, Sept. 2008.

[7] C. L. Zitnick and S. B. Kang. Stereo for image-based rendering using image over-segmentation. *Int. J. Comput. Vision*, 75(1):49–65, 2007.

People Search in Surveillance Videos

Daniel A. Vaquero
Four Eyes Lab, UCSB
daniel@cs.ucsb.edu

Rogerio S. Feris
IBM Research
rsferis@us.ibm.com

Lisa Brown
IBM Research
lisabr@us.ibm.com

Arun Hampapur
IBM Research
arunh@us.ibm.com

Matthew Turk
Four Eyes Lab, UCSB
mturk@cs.ucsb.edu

1. INTRODUCTION

In traditional surveillance scenarios, users are required to watch video footage corresponding to extended periods of time in order to find events of interest. However, this process is resource-consuming, and suffers from high costs of employing security personnel. The field of intelligent visual surveillance [2] seeks to address these issues by applying computer vision techniques to automatically detect specific events in long video streams. The events can then be presented to the user or be indexed into a database to allow queries such as "show me the red cars that entered a given parking lot from 7pm to 9pm on Monday" or "show me the faces of people who left the city's train station last week."

In this work, we are interested in analyzing people, by extracting information that can be used to search for them in surveillance videos. Current research on this topic focuses on approaches based on face recognition, where the goal is to establish the identity of a person given an image of a face. However, face recognition is still a very challenging problem, especially in low resolution images with variations in pose and lighting, which is often the case in surveillance data. State-of-the-art face recognition systems [1] require a fair amount of resolution in order to produce reliable results, but in many cases this level of detail is not available in surveillance applications.

We approach the problem in an alternative way, by avoiding face recognition and proposing a framework for finding people based on parsing the human body and exploiting part attributes. Those include visual attributes such as facial hair type (beards, mustaches, absence of facial hair), type of eyewear (sunglasses, eyeglasses, absence of glasses), hair type (baldness, hair, wearing a hat), and clothing color. While face recognition is still a difficult problem, accurate and efficient face detectors[1] based on learning approaches [6] are available. Those have been demonstrated to work well on challenging low-resolution images, with variations in pose and lighting. In our method, we employ this technology to design detectors for facial attributes from large sets of training data.

[1]The face detection problem consists of localizing faces in images, while face recognition aims to establish the identity of a person given an image of a face. Face detection is a challenging problem, but it is arguably not as complex as face recognition.

Our technique falls into the category of *short term recognition* methods, taking advantage of features present in brief intervals in time, such as clothing color, hairstyle, and makeup, which are generally considered an annoyance in face recognition methods. There are several applications that naturally fit within a short term recognition framework. An example is in criminal investigation, when the police are interested in locating a suspect. In those cases, eyewitnesses typically fill out a *suspect description form*, where they indicate personal traits of the suspect as seen at the moment when the crime was committed. Those include facial hair type, hair color, clothing type, etc. Based on that description, the police manually scan the entire video archive looking for a person with similar characteristics. This process is tedious and time consuming, and could be drastically accelerated by the use of our technique. Another application is on finding missing people. Parents looking for their children in an amusement park could provide a description including clothing and eyewear type, and videos from multiple cameras in the park would then be automatically searched.

In this abstract, we present our implementation of the first video surveillance system with retrieval capabilities based on people's fine-grained parts and attributes. The system receives queries such as "show me the people entering the CS building on Monday, who were wearing a red shirt, sunglasses and a hat," and retrieves the corresponding videos.

2. SEARCH BY PARTS AND ATTRIBUTES

In [5], we propose a general framework for searching for people in surveillance data, comprising three main elements: sensors, body parts, and their attributes. In this section, we summarize our implementation, which considers a particular case of the general framework. The sensor is a low-resolution (320x240) color video camera. The body parts we localize are: face, torso and legs. We divide the face region into three subregions: upper part, middle part, and lower part. The following attributes are then extracted from each body part: hair type (bald, hair, wearing a hat), from the upper face region; eyewear type (sunglasses, eyeglasses, absence of glasses), from the middle face region; facial hair type (beard, mustache, absence of facial hair), from the lower face region; and dominant color, from the torso and legs.

To detect faces in images, we use a cascade of Adaboost classifiers [6], trained from image patches (of 20x20 size)

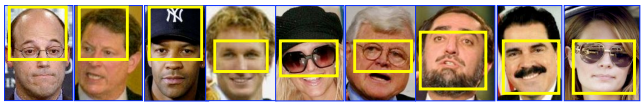


Figure 1: Examples of labelings for each facial attribute (bald, hair, hat, no glasses, sunglasses, eyeglasses, beard, mustache, no facial hair).

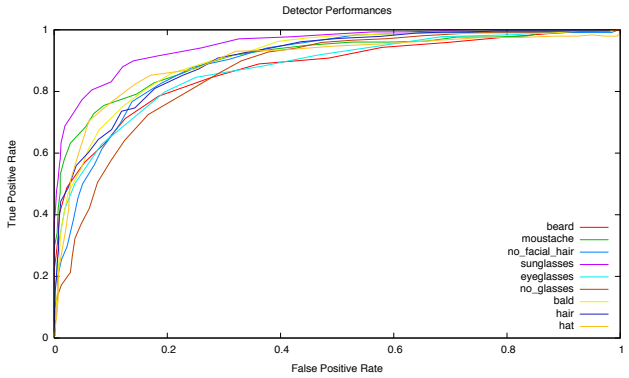


Figure 2: Receiver Operating Characteristic curves for the facial attribute detectors, evaluated on a set of face images collected from the Internet (best seen in color[2]).

containing faces. Given the detected face's position, the locations of the torso and legs are estimated as rectangles with fixed aspect ratio, whose position and size are roughly estimated from the position and size of the detected face. Since people have different heights and body shapes, the rectangles may not precisely outline the regions of the torso and the legs; however, this is not a problem, as for the sake of this implementation we are interested only in obtaining the dominant color attribute for these parts.

To extract facial attributes, we have trained nine additional Viola-Jones detectors [6], one for each attribute. A dataset of about 9,800 frontal face images (with slight pose variations) collected from the Labeled Faces in the Wild dataset [3] and from Internet sites has been created for this purpose. The images have been manually labeled by indicating bounding boxes for specific attributes (Figure 1).

The Viola-Jones attribute detectors are applied to the corresponding face regions, in order to extract facial attributes. To determine the color of the torso and legs, we classify the corresponding regions into one of 8 colors – red, green, blue, yellow, orange, purple, black, and white – by quantizing each pixel in the region into one of these colors [4] and then choosing the dominant color.

3. EXPERIMENTAL RESULTS

We performed two sets of experiments: evaluation of the facial attribute detectors on face images collected from the Internet, and deployment of our surveillance system in a real-world scenario (at IBM). Figure 2 displays the ROC curves

[2]An electronic version of this abstract is available at www.cs.ucsb.edu/~daniel

Figure 3: First search results obtained from queries for "bald people" (top 3 rows) and "red shirts" (bottom 3 rows, best seen in color[2]).

for all detectors, evaluated on a set of 2,698 Internet images. The plots were obtained by varying the detection threshold. This experiment demonstrates that machine learning techniques in conjunction with large amounts of training data can be used to design reliable facial attribute detectors.

To illustrate the results in the surveillance scenario, Figure 3 shows two query results, one for "bald people," and another for "red shirts." In [5], we present a more detailed quantitative analysis and discuss failure cases, such as the presence of shadows that resemble beards and mustaches, and the lack of resolution to appropriately capture the frames of the eyeglasses, making them nearly impossible to detect even for human observers. We also show promising initial results on the use of thermal infrared imagery as a solution to these issues, as this modality reduces illumination effects, facilitates the detection of transparent eyeglasses, and accentuates skin/hair contrast.

4. REFERENCES

[1] Face Recognition Vendor Test. http://www.frvt.org/.

[2] W. Hu, T. Tan, L. Wang, and S. Maybank. A survey on visual surveillance of object motion and behaviors. *IEEE Trans. Syst., Man, Cybern., Pt. C*, 34(3):334–352, 2004.

[3] G. Huang, M. Mattar, T. Berg, and E. Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. In *ECCV Workshop on Faces in Real-Life Images*, 2008.

[4] D.-C. Tseng and C.-H. Chang. Color segmentation using UCS perceptual attributes. In *Proc. Natl. Sci. Council: Part A*, volume 18, pages 305–314, 1994.

[5] D. A. Vaquero, R. S. Feris, D. Tran, L. Brown, A. Hampapur, and M. Turk. Attribute-based people search in surveillance environments. In *IEEE Workshop on Applications of Computer Vision (WACV)*, 2009.

[6] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *IEEE Conf. on Comp. Vision and Pattern Recognition (CVPR)*, 2001.

Combinatorial Optimization under Uncertainty

Pegah Kamousi
Department of Computer Science
UC Santa Barbara
pegah@cs.ucsb.edu

Subhash Suri
Department of Computer Science
UC Santa Barbara
suri@cs.ucsb.edu

ABSTRACT

Consider a probabilistic graph $G = (V, E)$, in which node i is only present with probability p_i and absent otherwise, or a set of points in a normed space where each point appears with probability p_i. We consider some combinatorial optimization problems in such probabilistic settings. We prove that some optimization problems, like the expected area of the convex hull of points remain in **P** as in deterministic case, but some problems, like the expected length of the minimum spanning tree (MST), become **#P**-hard. We will specially focus on the probabilistic minimum spanning tree problem (PMST hereafter), prove some hardness results, and provide exact and approximate solutions for some special cases, such as metric graphs and Euclidean graphs, and graphs of bounded treewidth.

1. INTRODUCTION

Networks have become widely used for modeling complex systems which are subject to component failures. We often seek to optimize different costs associated with such networks, such as the cost of connecting certain terminals. Combinatorial optimization has undoubtedly been one of the most exciting and rapidly growing fields in discrete mathematics. Probabilistic Combinatorial Optimization Problems (PCOPs) are generalized version of combinatorial optimization problems, with explicit inclusion of probabilistic elements in problem definition. The motivation behind inclusion of such probabilistic elements is to make the model suitable for real world applications in which randomness is a major concern. There are also several applications of PCOPs in strategic planning for communication and transportation services, job scheduling, etc. All previous work in such frameworks has either focused on asymptotic analysis of problems when the costs are drawn from a distribution (see [1] for example), or on *A priori Optimization* (see [3, 5]).

Consider a graph $G = (V, E)$, where nodes or links might fail with certain probabilities. In absence of such failure probabilities, we can easily count the number of connected components, find the lengths of the shortest paths or find the MST of the graph. With inclusion of probabilistic elements, however, the application of existing algorithms becomes, in most cases, impossible and we need new techniques to treat them. We will discuss some problems in this setting, compare their intrinsic hardness to their deterministic counterparts and provide some techniques to solve or approximate them. Our focus will be on the minimum spanning tree problem, which we will prove to be #-P-hard to solve exactly, even for the case G is a Euclidean graph induced by a set of points in spaces of dimension greater than 2. We will provide a $log(n)$-approximation for metric graphs, and a constant factor approximation for Euclidean graphs in spaces of any dimension. We also prove that PMST is NP-hard, even to approximate within a constant, for general graphs. Finally, we show that the problem can be solved in polynomial time for the special case of graphs of bounded treewidth.

2. THE PMST PROBLEM

Given a complete graph $G = (V, E)$, a rational probability $0 \leq p_i \leq 1$ of being present for each $v \in V$, and a weight $w(e) \in \mathcal{Z}$ for all $e \in E$, the PMST problem asks to compute the expected weight of the MST, where the expectation is over the subsets of points being present. We will prove that this problem is #P-hard. The class of #P-hard problems was introduced by Valiant [6], where he showed that the classical network reliability problems (NRP) are #P-hard. In that problem, we are given a graph $G' = (V', E')$, and a probability q_i of failure for each vertex, and the goal is to compute the probability that a subset $A \subseteq V$ of points or a pair (s, t) of terminals are connected via the surviving subgraph of G'.

2.1 The complexity of the PMST problem

In a special case of NRP, which is still #P-hard, $q_i = 1/2$ for all i, and we wish to count the number of subsets $S \subseteq V$ such that a give pair (s, t) of terminals are connected via the subgraph induced by S. This problem is called the $S - T$-Connectedness problem.

THEOREM 1. *The PMST problem is in* **P**, *if and only if the* $S - T$*-Connectedness problem is in* **P**.

We will prove this by reducing the above problems to each other in polynomial time. This proves the PMST problem to be #P-hard. We also prove that this problem is NP-hard, even to approximate within a constant factor.

In the well known NP-hard Steiner subgraph problem, we are given a graph $G = (V, E)$, and a subset $A \subseteq V$, and wish to find the minimum subgraph of G that connects all the nodes in A.

THEOREM 2. *If we can approximate, in polynomial time, the PMST within any constant, then we can solve, in polynomial time, the Steiner subgraph problem for any graph* G.

The above proofs all rely on arbitrary edge weights, which renders it invalid for the geometric case, i.e., for the case of

a Euclidean graph over a set of points in a Euclidean space, where for each pair (i, j) of vertices the weight of the edge connecting them is their Euclidean distance. We prove that the problem remains hard even in that case.

THEOREM 3. *Given a graph G for which we wish to solve the NRP, we can construct, in polynomial time, a set S of points in the 3-dimensional Euclidean space, such that by computing the expected weight of the MST of G', the induced Euclidean graph of S, we can solve, in polynomial time, the NRP for G.*

2.2 Approximating PMST for Metric Graphs

In this section we provide an algorithm which approximates the PMST, within a factor of $\log(n)$, where n is the number of nodes in the graph. Let us arbitrarily rank the nodes, and for each node, compute the **expected** weight of the edge connecting it to the closest node with a lower rank. Consider the point with rank i. Let $C_i(j)$ be the probability that point j is present and none of the points closer to i than j are present. Then

$$C_i(j) = p_j . \prod_{k<i, d(k,i)\leq d(j,i)} q_k,$$

and the expected length of the edge connecting i to previous points is $E[e_i] = p_i . \sum_{j<i} C_i(j) d(i, j)$. Therefore

$$E[MST] = \sum_{i \in P} E[e_i].$$

For the deterministic case, the counterpart of the above algorithm would be to process the nodes in an arbitrary order, and connect each node to the closest nodes among the previous ones. This algorithm approximates the Steiner tree of a metric graph by a factor of $O(\log n)$, which implies that it is also a $O(\log n)$-approximation for MST. A simple proof can be found in [2].

2.3 Approximating PMST for Euclidean Graphs

In this section, we provide a constant factor approximation for PMST, for a Euclidean graph of a set of points on the plane. This method can be extended to Euclidean spaces of any dimension.

The Yao graph of a set V of points on the plane was defined by Yao in [7]. Given any number $k \geq 6$, at each node u, k equally-separated rays originating at u define k cones. Connect each node u to its closest neighbor in each of the cones, to obtain the Yao graph of points. Let $w(MST)$ be the weight of the minimum spanning tree. It is well known that the MST is a subset of the Yao graph, but in general the weight of the Yao graph can be $O(n)$ times greater than $w(MST)$. See Figure 1 for an example of a Yao graph.

A graph is called light-weight if its weight is $O(w(MST))$. In the following, we propose a way of pruning the Yao graph, such that the resulting subgraph, still contains the MST and is light-weight, and show how this approximation algorithm can be adapted to the case the points will appear probabilistically.

THEOREM 4. *Any subgraph G of the Yao graph is light-weight if for any edge $uv \in G$, uv does not belong to any cycle of length 4 in the original complete graph such that uv is the longest edge in that cycle.*

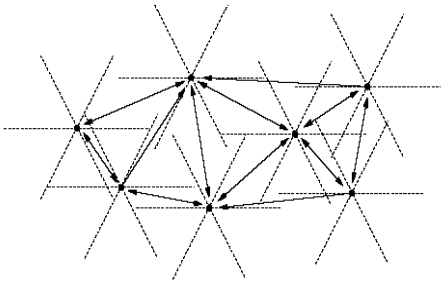


Figure 1: An example of a Yao Graph

The proof uses the *Isolation Property*, as defined in [4]. For each edge uv, let $Y(u, v)$ be the probability that uv belongs to the Yao graph, and $F(u, v)$ be the probability that it does not belong to any cycle of length 4 such that uv is the longest edge in that cycle, given that it does belong to the Yao graph. Then we have

$$\sum_{i,j} Y(i, j).F(i, j).\|ij\| = O(w(MST)).$$

We will show how each of the quantities Y(u,v) and F(u,v) can be obtained in polynomial time. The running time of the algorithm will be $O(n^3)$.

3. OTHER PROBABILISTIC PROBLEMS

We have also considered other problems in the probabilistic setting. One of the problems is computing the area of the convex hull. Suppose we have a set P of points on the plane where each point appears with probability p_i, and the goal is to compute the expected area, or expected perimeter of the convex hull of P. We will show that this problem, unlike the MST problem, remains in **P** in probabilistic case. We can prove that the probabilistic shortest path problem and the Probabilistic Maximum Spanning Tree (PMXST) are hard for general graphs. What remains open is the real complexity of the PMST and PMXST on the plane.

4. REFERENCES

[1] D. Aldous and J. M. Steele. Asymptotics for euclidean minimal spanning trees on random points. *Probability Theory and Related Fields*, 92(2), 1992.

[2] N. Alon and Y. Azar. On-line steiner trees in the euclidean plane. In *SCG '92: Proceedings of the Eighth annual Symposium on Computational Geometry*, pages 337–343, New York, NY, USA, 1992. ACM.

[3] D. J. Bertsimas, P. Jaillet, and A. R. Odoni. A priori optimization. *Oper. Res.*, 38(6):1019–1033, 1990.

[4] G. Das, G. Narasimhan, and J. Salowe. A new way to weigh malnourished euclidean graphs. In *SODA '95: Proceedings of the sixth annual ACM-SIAM symposium on Discrete algorithms*, pages 215–222, 1995.

[5] P. Jaillet. Analysis of probabilistic combinatorial optimization problems in euclidean spaces. *Math. Oper. Res.*, 18(1):51–70, 1993.

[6] L. G. Valiant. The complexity of enumeration and reliability problems. *SIAM Journal on Computing*, 8(3):410–421, 1979.

[7] A. C. Yao. On constructing minimum spanning trees in k-dimensional spaces and related problems. Technical report, Stanford, CA, USA, 1977.

Energy Conservation in Datacenters through Cluster Memory Management and Barely-Alive Servers

Vlasia Anagnostopoulou*, Susmit Biswas*, Alan Savage*,
Ricardo Bianchini†, Tao Yang*, Frederic T. Chong*

*Department of Computer Science, University of California, Santa Barbara
†Department of Computer Science, Rutgers University

ABSTRACT

This abstract is about a new, energy-efficient power-state for servers in datacenter clusters, and a global cache management scheme to accommodate for this new state, while managing wisely all of the cluster's memory. With this new state, which we call Barely-Alive, we anticipate very encouraging power savings.

1. INTRODUCTION

Energy represents a large fraction of the operational cost of Internet services, which run in large clusters in datacenters. As a result, previous works have proposed approaches for conserving energy across these clusters, such as consolidating workloads into a subset of servers and turning others off [2, 3, 7, 8] and leveraging dynamic voltage and frequency scaling [3, 4, 5].

Consolidation is particularly attractive for two reasons. First, current resource provisioning schemes in datacenters leave server utilizations under 50% almost all the time [5]. At these utilizations, server energy efficiency is very low [1]. Second, current servers consume a significant amount of energy even when they are completely idle [1]. Despite its benefits, services typically do not use this technique. A major reason is the fear of slow response times during re-activation in handling traffic spikes. Another reason is that services want to maximize the amount of main memory available for data caching across the server cluster, thereby avoiding disk accesses or content re-generation.

In this abstract, we propose an approach that does not completely shutdown idle servers, keeps in-memory application code/data untouched, and allows free memory space to be used for cooperative application data caching. Specifically, we propose to send servers to a new "barely-alive" power state, instead of turning them completely off after consolidation. In barely-alive state, a server's memory can still be accessed, even if many of its other components are turned off. The new state does not affect the consolidation algorithm, which can stay the same as before [2, 7].

To manage the memory itself, we also design a distributed middleware that transforms the servers' main memories into a large cooperative cache. The main contributions of the middleware are (1) its ability to accommodate barely-alive servers; and (2) its ability to dynamically re-size the amount of cache space across the cluster to the minimum required to respect the SLA (or equivalently to achieve a desired average hit ratio). The sizing of the cache is based on a distributed implementation of a Stack algorithm [6]. Any memory that is not in use by the middleware can be used by applications.

Our preliminary evaluation uses trace-driven simulation of a server cluster running a search engine application. We model the application using data from AOL and Ask.com. Our results show that we can conserve up to 24% energy and 49% power for this application. Moreover, our preliminary results show that the middleware is capable of properly sizing the cluster-wide cache to produce a desired cache hit ratio.

The remainder of this abstract is organized as follows. Next, we discuss the details of the barely-alive power state. In the following section, we present the main algorithm of the cache manager, and after this, we conclude the abstract.

2. BARELY-ALIVE SERVERS

Services would like to conserve server energy during periods of less-than-peak load using consolidation and server turn off. However, servers cannot typically be turned off, since their re-activation can produce high response times (e.g., due to operating system reboot) during traffic spikes and their main memories are required for data caching. Furthermore, the servers may need to stay on for the service to achieve a higher aggregate disk throughput.

Given these constraints, we propose the barely-alive power state. Servers can be sent to barely-alive state, instead of off state, after the workload is consolidated on another set of servers. A barely-alive server allows remote access to its main memory, even though most of its other components are turned off to conserve energy. Because the memory contents (including the operating system) are not affected, transitioning to and from barely-alive state can be very fast.

With respect to the processing of memory accesses in barely-alive state, we envision two possible approaches. The first leaves one of the cores active and responsible for receiving and performing accesses to the server's main memory. *Because all accesses are performed by the host processor, all memory addressing can be done using virtual addresses.* This approach does not require changes to current multi-core hardware, besides the ability to turn off cores independently.

Unfortunately, as the number of cores per CPU increases with each processor generation, it is less likely that we will be able to manage the cores' power states independently.

As a result, we will not be able to leave a single core active, increasing the energy consumption of the barely-alive state. Thus, our second approach, the one we actually evaluate in this paper, turns off all the cores but keeps the memory controller on (even if the controller is on chip). Remote memory accesses occur through a very low-power embedded processor built into the network interface. This processor accesses memory by driving the memory controller directly, just as in regular DMA operations involving the network interface. *Because the embedded processor does not understand virtual addresses, the remote memory accesses have to specify physical addresses or be translated to physical addresses in software. (Our middleware takes the latter approach.)*

Obviously, the embedded processor has to implement some sort of (RDMA) communication protocol to be able to receive memory access requests coming from active servers and reply to them. As our target system is a server cluster, this communication protocol can be lean and simple. Because the barely-alive state is essentially independent of this protocol, we do not discuss it further.

Regardless of the exact implementation of the barely-active state, the consolidation algorithm can be the same as before. In other words, servers can be transitioned from active to barely-alive state and back exactly at the same points as a standard consolidation algorithm would turn them off and on, respectively.

3. CLUSTER MEMORY MANAGEMENT

A key aspect of our design is the ability of the middleware to re-size the cooperative cache. The goal is to use just enough memory for the cache as required by the service's SLA. In fact, we assume that the SLA requirement can be translated into a hit-ratio requirement. Shrinking the cache to this minimum size can be critical in the context of consolidation, since more memory can be used by other applications. A second goal is to size all local caches uniformly, so that we are eventually able to manage the caching of data from multiple applications with a simple scheme. As aforementioned, our approach to sizing the cooperative cache relies on a distributed implementation of the stack algorithm.

The stack algorithm predicts the hit-ratio of a cache hierarchy, given its capacity, the replacement policy, and a sequence of memory accesses. Precisely, using a single pass over the stream of memory accesses, it computes the hit-ratio that would be achieved by all cache sizes. In our system, we consider a single-level memory hierarchy, i.e. main memory, and use LRU as the replacement policy as in [6] (see first part of figure below).

In the cache manager implementation, each application using the middleware has its own set of stacks. Servers communicate at fixed time intervals (time windows), e.g. every hour. During each round of communication, each server broadcasts the local cache size that it would require to achieve the desired hit ratio. With information from all nodes, each server can compute the average of the required local cache sizes and reconfigure its local cache to that size. The local stack information is then reset (figure below).

The server daemon maintains the local stack information of

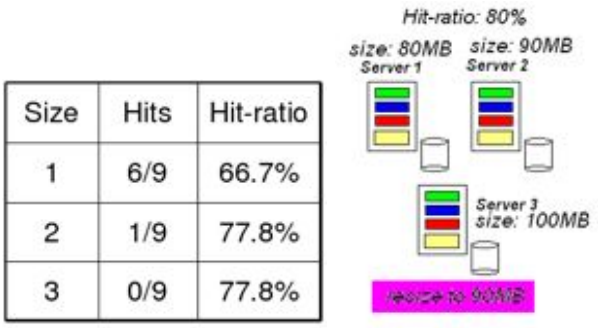


Figure 1: Cache manager operation

each application separately. Object fetches update the local stack in the obvious way. Object stores are handled by inserting new entries into the stack, whereas object invalidations are handled by removing entries from the stack.

4. CONCLUSION

In this paper, we sought to make consolidation and server turn off more practical. With that in mind, we made two main contributions. First, we proposed a new power state, called barely-alive. When a server is in this state, its memory can still be accessed, even though the entire CPU is turned off. Second, we proposed a middleware for cooperative caching that accommodates barely-alive servers. The middleware also dynamically re-sizes the cache, according to a desired hit ratio, using a distributed stack algorithm. Our preliminary results showed that our proposed caching middleware and barely-alive power state offer promising energy savings and the mechanisms to facilitate cluster-wide memory management for application-level caching.

5. REFERENCES

[1] L. A. Barroso and U. Holzle. The Case for Energy-Proportional Computing. *IEEE Computer*, 40(12), 2007.

[2] J. Chase et al. Managing Energy and Server Resources in Hosting Centers. In *Proceedings of SOSP*, October 2001.

[3] Y. Chen et al. Managing Server Energy and Operational Costs in Hosting Centers. In *Proceedings of SIGMETRICS*, June 2005.

[4] M. Elnozahy, M. Kistler, and R. Rajamony. Energy-Efficient Server Clusters. In *Proceedings of PACS*, 2002.

[5] X. Fan, W.-D. Weber, and L. A. Barroso. Power Provisioning for a Warehouse-sized Computer. In *Proceedings of ISCA*, June 2007.

[6] R. L. Mattson et al. Evaluation Techniques for Storage Hierarchies. *IBM Systems Journal*, 9(2), 1970.

[7] E. Pinheiro et al. Load Balancing and Unbalancing for Power and Performance in Cluster-Based Systems. In *Proceedings of COLP*, September 2001.

[8] K. Rajamani and C. Lefurgy. On Evaluating Request-Distribution Schemes for Saving Energy in Server Clusters. In *Proceedings of ISPASS*, March 2003.

Profile Based Sub-Image Search in Image Database

Vishwakarma Singh
University of California, Santa Barbara
vsingh@ucsb.edu

Ambuj K. Singh
University of California, Santa Barbara
ambuj@ucsb.edu

ABSTRACT

This paper proposes a new feature vector called *profile* based on neighborhood profiling that captures the spatial geometry of keypoints in an image. Profile computed for each keypoint in a bag of words representation of an image is effective for sub-image search. Search using profiles is a single-phase process requiring no geometric validation, yields high precision on natural images, and needs a small visual codebook. The proposed search technique differs from traditional methods, which first generate a set of candidates disregarding spatial information and then verify them geometrically. Conventional methods also use large codebooks. We achieve a precision of 81% on a combined data set of synthetic and real natural images for top-10 queries; that is 31% higher than the common candidate generation approach.

1. INTRODUCTION

Community contributed image sites (e.g. Flickr.com) and stock photo sites (e.g. Gettyimages.com) have seen an unprecedented growth in the recent times. Search of images by example is one of the common tasks performed on these data sets. A related task is sub-image retrieval [4]. Repositories contain images of objects taken under varying imaging conditions having affine, viewpoint, and photometric differences and also varying degrees of occlusion. Local features like SIFT [2, 3] are used in literature [4] with fair success to measure similarity between images. Images are scanned to detect keypoints, covariant regions are extracted around each point, and finally a high dimensional local feature vector [2] representation of each region is obtained [3].

(a) Query (b) Result

Figure 1: A pathological case. False match is obtained in the top-5 results because spatial relationship is not considered.

Researchers have pursued two-phase techniques to retrieve similar images using local descriptors. The first phase consists of candidate generation disregarding spatial relationships between points and the second phase consists of geometric verification. Candidates are generated using two common approaches. One approach [4] transforms each image into a bag of visual words represented as a histogram by clustering all the feature vectors. Another approach [2] finds top-k nearest neighbors of each keypoint feature vector of the query image in the dataset of local feature vectors of all the images. Both approaches are made efficient by using indexing techniques [1].

We present a pathological result of sub-image search performed using the previously described first-phase methods. Figure 1(a) is a query sub-image and Figure 1(b) is an image in the top-5 results retrieved from the dataset. We can see that the components of the query pattern are scattered randomly in Figure 1(b) and it is not a meaningful match. This motivates the necessity of using spatial relationships between matched points in a sub-image search. Existing literature uses geometrical verification (Hough [2], Ransack [4]) on the candidate images to find the best match or re-rank the top-k candidates. Generating a small set of high quality candidates is very important for all conventional approaches to reduce the cost of geometric verification.

In this paper, we present a new feature vector which produces high quality results for sub-image search without geometric verification using a single-step search and a small codebook.

2. PROFILE CREATION

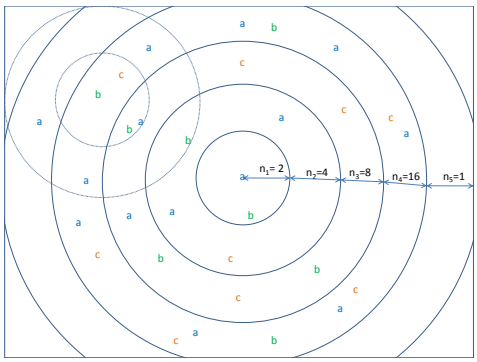


Figure 2: The profile of a point is defined by a set of histograms. Each histogram summarizes the points in a ring. The number of points in a ring doubles until the last ring.

In this section, we design our novel profile based feature vector. We draw concentric circles as shown in Figure 2 around each keypoint p in an image. Each ring is represented as a histogram h of visual words of keypoints lying in it. Profile $\mathbf{H} = (h_1, h_2, \cdots, h_m)$

Figure 3: Top-5 results for 2 real queries over general image dataset using profile based search.

of a point p, where m is the number of rings, is a concatenated list of ring histograms ordered from the center towards the outer rings. The number of points in a ring is defined by $Size(h_i) = 2 * Size(h_{i-1})$. The number of points in the first ring is n_0 and the last ring contains the left over points.

Similarity between two profiles H_i and H_j, where the number of rings in H_i is m and in H_j is n respectively, is given by

$$Sim(H_i, H_j) = \sum_{k=0}^{min(m,n)} e^{-\lambda k} * S_k$$

Here, λ is a decaying parameter learned by evaluation. Similarity between corresponding ring histograms, S_k, can be computed using Jaccard's Coefficient or Cosine measure. We use Jaccard's Coefficient. Distance between profiles is computed as a complement of the similarity. The maximum value of similarity between two corresponding histograms is 1. Therefore, maximum similarity between two profiles is

$$Sim_{max}(H_i, H_j) = \sum_{k=0}^{min(m,n)} e^{-\lambda k} * 1$$

Sub-Image Search: All the images represented as bags of words are converted into bags of profiles. Query Q is also processed to generate a bag of profiles. The best matching sub-image for a given query is the region around the keypoint corresponding to the best matching profile in the dataset. If Q has m profiles and N is the total number of profiles in the dataset then the best matching sub-image is the region around ith profile where i is obtained by.

$$\arg_i \max_{j \leq m} \left\{ \max_{i \leq N} Sim(H_{q_j}, H_i) \right\}$$

3. EVALUATION

We downloaded natural images from Flickr. Our dataset has 1000 images (800-natural and 200-synthetic). We used a total of 52 queries (35-natural and 17-synthetic). We created bag of word srepresentation from the SIFT feature vectors by picking 500 random centers using k-means clustering. For creating profiles, we chose n_0=50 points in the first ring. We used λ=1/3 as decaying parameter for aggregation.

Precision Test: We computed precision of top-k results obtained using our approach and compared it against the precision of the top-k candidates retrieved by conventional methods which do not take spatial relationships into account. We used Cosine, L_1, and Jaccard's Coefficient as measure for conventional methods. We considered both the schemes of standard tf-idf weighting and without tf-idf weighting for candidate generation approach. We do not consider tf-idf weighting for our profile based approach. Figure 4 and Figure 5 show the comparative precision of various methods for the non-weighted and the weighted cases, respectively. We find that our approach yields 81% precision rate which is 31% better than the best technique for top-10 results in both the schemes. We also experimented with varying λ and achieved 20% better precision for $\lambda \leq 1$. We also experimented by weighting the symbols with the area of covariant regions but achieved less precision.

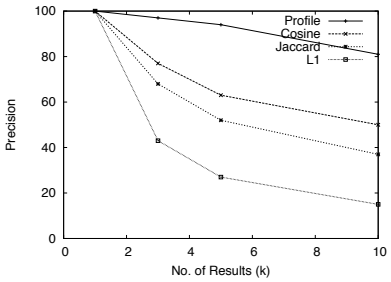


Figure 4: Comparison of conventional methods, using different distance measures without tf-idf weighting, with profile based approach.

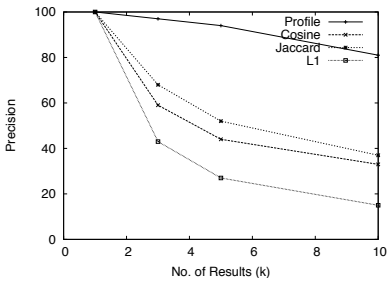


Figure 5: Comparison of conventional methods, using different distance measures and tf-idf weighting, with profile based approach.

Visual Results: We present top-5 visual results for 2 real queries from our search. Our profile based approach retrieves high quality results irrespective of varieties of noise present in the dataset. We outline the matching sub-region in a result image with red box.

4. REFERENCES

[1] A. Gionis, P. Indyk, and R. Motwani. Similarity search in high dimensions via hashing. In *VLDB*, 1999.
[2] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60:91–110, 2004.
[3] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. V. Gool. A comparison of affine region detectors. *IJCV*, 65.
[4] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *CVPR*, pages 1–8, 2007.

Generalized Private Querying on Public Data in Single Client-Server Settings

Shiyuan Wang
Department of Computer Science, UC Santa Barbara
sywang@cs.ucsb.edu

Divyakant Agrawal
Department of Computer Science, UC Santa Barbara
agrawal@cs.ucsb.edu

Amr El Abbadi
Department of Computer Science, UC Santa Barbara
amr@cs.ucsb.edu

ABSTRACT

We consider the problem of general private queries on public data in single client server interactions, where a client executes queries on the server but does not want to reveal the query to the server. We propose a generalized private querying protocol called *Bounding-Box PIR* (*bb*PIR) which is practical and can be applied in large scale information services while permitting clients to specify the privacy level and private service charge budget they desire.

1. INTRODUCTION

We consider the problem of *private querying on public data*, in which a client executes queries on the data of a public server with its private data as the filtering condition, while not revealing to the server the exact values of the private data in the query. A general and promising use is in personalized search and recommendation subscriptions through big internet information service providers such as Google, Yahoo and Microsoft Bing. Users need these public services in their daily lives. However, users are concerned that their private profile data or their personal tastes might be disclosed or compromised through analysis or inferences.

In such applications, the server owns a huge amount of data which can be of interests to different clients, while a client has a small amount of private data that she may use for information retrieval. To apply private querying on public data in the general service scenarios, we have the following desiderata for a private querying solution: First, it should be *practical*, meaning that the solution should minimize the communication overhead between clients and the server per public data item requested and the computation overhead, especially for the server to support more clients. According to this desiderata, the solutions that transfer all public data to the client are infeasible. Second, for business profits, the solution should enable the server to charge the client based on the size of information exposed to the client. Note that this amount could be larger than the size of the real answer to the query, because the query is blurred to protect the privacy of the client. The client should be able to specify *flexible privacy* and a *charge budget* as she needs.

Unfortunately, few studies have addressed the general problem of private querying on public data in single server settings, and even fewer can be adapted to satisfy the above desiderata. The two closest studies are k-*Anonymity* [3] and *Computational Private Information Retrieval* (*c*PIR) [1]. k-Anonymity generalizes data values into ranges or partitions such that each value is indistinguishable among at least k values in one range or partition. However, k-Anonymity may breach privacy, if an anonymized range or partition is too small, where the private value is too similar to other values. Computational Private Information Retrieval (*c*PIR) [1] retrieves a bit from a public bit string without revealing to the server the position of the desired bit. It achieves computationally complete privacy by using expensive operations over the entire public data. Because of the expensive computation cost, even the *c*PIR technique with the least expensive operation, modular multiplication [1], is criticized as up to orders of magnitude less efficient than simply transferring the entire data from the server to the client [2].

To achieve the above mentioned desiderata, we propose a generalized approach called *Bounding-Box PIR* (*bb*PIR) that unifies both k-Anonymity and *c*PIR based private querying approaches. *bb*PIR is based on *c*PIR to achieve better privacy than k-Anonymity, and at the same time relax the complete privacy of *c*PIR for server performance gain.

2. BACKGROUND ON *c*PIR

*c*PIR relies on the computational intractability of the *Quadratic Residuosity* problem. Let N be a natural number, and $Z_N^* = \{x | 1 \leq x \leq N, gcd(N,x) = 1\}$. x is a *quadratic residue* (QR) mod N if $\exists\ y \in Z_N^*$ s.t. $y^2 = x \bmod N$. Otherwise, x is a *quadratic nonresidue* (QNR) mod N. The problem is considered computationally most difficult if $N = p_1 p_2$, where p_1 and p_2 are two large distinct primes with equal number of bits, $m/2$. Let $Z_N^{+1} = \{y \in Z_N^* | (\frac{y}{N}) = 1\}$. The *Quadratic Residuosity Assumption* (QRA) says that for $y \in Z_N^{+1}$, without knowing p_1 and p_2 in advance, the probability of distinguishing y between a QR and a QNR is negligible for large enough number of bits m [1]. The problem is much easier if p_1 and p_2 are known.

*c*PIR [1] utilizes the computational difference in determining whether a number is QR or QNR if p_1 and p_2 are known or not. Let n be the total number of public data items (bits in this case). The public data is organized into an $s \times t$ matrix M (choose $s = t = \sqrt{n}$). Let (e, g) be the two dimensional address of the bit entry queried by the client. The *c*PIR protocol is as follows:

1. Initially, the client sends to the server an m-bit number N which is the product of two random $m/2$-bit primes p_1 and p_2.
2. The client generates a vector of t m-bit random numbers in Z_N^{+1}, $y = [y_1, ..., y_t]$, s.t. y_g is a QNR and all other y_i ($i \neq g$) are QR. It sends the vector y to the server.
3. The server computes for each row i of M a modular product $z_i = \Pi_{j=1}^{t} w_{i,j}$, where $w_{i,j} = y_j^2$ if $M_{i,j} = 0$, and $w_{i,j} = y_j$ if $M_{i,j} = 1$.
4. The server sends to the client $z_1, ..., z_s$.
5. The client determines that $M_{e,g} = 0$ if z_e is QR, and $M_{e,g} = 1$ if z_e is QNR.

Fig. 1 illustrates a simple example of how to use *c*PIR to retrieve a bit at $M_{2,3}$. The server can not figure out if a y_i or z_i is QR or QNR, because the server does not know p_1 and p_2, but the client can. Since all z_i are available to the client, the client can get all $M_{i,g}$ ($1 \leq i \leq s$) bits in column g, and hence the public data in all rows, s, are exposed to the client.

3. DATA MODEL

Assume the public data is stored in an $s \times t$ matrix M, where each entry is a b-bit data item. Each public data item x has a numeric

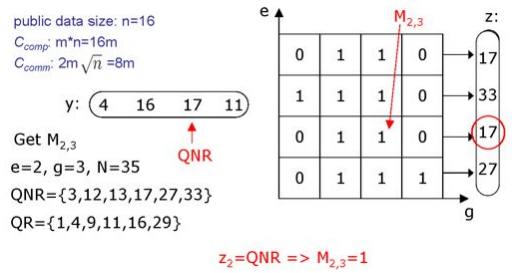


Figure 1: *c*PIR Example

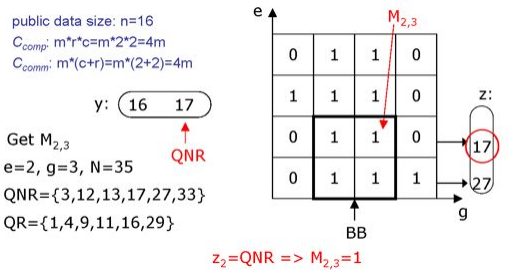


Figure 2: *bb*PIR Example: b=1

key KA which identifies the address of x in M. Sorting the public data by KA in ascending order, the public data items are put in M, columnwise from the leftmost column to the rightmost column.

The client specifies her privacy requirement and charge budget as (ρ, μ), where ρ is the highest allowed probability that the server can identify a requested item (e.g. $\rho = 1/k$ in k-Anonymity), and μ is the charge limit for one item (the upper bound of the number of items retrieved for one requested item). We keep track of four important metrics: (1) *Communication Cost* C_{comm}, the cost of communication between the client and the server in terms of the number of bits. (2) *Computation Cost* C_{comp}, server computation cost in terms of the number of involved public data bits. (3) *Privacy Breach Probability* P_{brh}, the probability that the server can figure out a requested item. (4) *Server Charge* C_{srv}, the number of interpretable public data items retrieved from the server.

4. BOUNDING-BOX PIR

The basic idea of Bounding-Box PIR (*bb*PIR) is to use a bounding box BB (an $r \times c$ sub-matrix on M) as the blurred range around the item x requested by the client, and then apply *c*PIR on the bounding box. *bb*PIR will find an appropriately sized bounding box that satisfies the privacy request ρ for each retrieved item, and achieves overall good performance in terms of Communication and Computation Costs without exceeding the Server Charge budget μ for each retrieved item.

Since *bb*PIR operates on an $r \times c$ sub-matrix of M instead of the entire matrix M as in *c*PIR, its client query y is a vector of c m-bit numbers, its server answer z are vectors of r m-bit numbers, and m-bit modular multiplication is applied on all the b-bit data items in the sub-matrix. Therefore, C_{comm}(*bb*PIR) is related to mc and mr. C_{comp}(*bb*PIR) is proportional to the area of the bounding box $mbrc$. P_{brh}(*bb*PIR) is related to $1/(rc)$. C_{srv}(*bb*PIR) is related to r.

We focus on a simple selection query that retrieves a public data item that matches a private key provided by client. Let us assume the client knows the exact address of the requested item, (e, g). Here two constraints from the client's requirement are Privacy Breach Probability $P_{brh} \leq \rho$, and Charge $C_{srv} = r \leq \mu$. Choose BB to be the minimum bounding box that is sufficient to satisfy the privacy breach limit ρ. So its area $|BB| = rc = \lceil 1/\rho \rceil$, and the minimum Computation Cost is $C_{comp} = mbrc = mb\lceil 1/\rho \rceil$. Then the goal is to minimize the Communication Cost $C_{comm} = mc + mbr$ without exceeding the charge limit μ, which is equivalent to minimizing $c + br$. The protocol for private querying on one item, denoted as *bbPIR-1item*, is described as follows:

1. Initially, the client sends to the server the size of the bounding box BB with area $\lceil 1/\rho \rceil$. The number of rows r and the number of columns s in the corresponding sub-matrix to BB are decided as follows:
   If $\mu \geq \lceil \sqrt{1/(\rho b)} \rceil$, set $r = \lceil \sqrt{1/(\rho b)} \rceil$, $c = \lceil \sqrt{b/\rho} \rceil$
   Otherwise, set $r = min(\mu, s)$, $c = min(\lceil 1/(\rho r) \rceil, t)$
2. To retrieve entry (e, g) in M, the client first locates the coordinates of BB on M with the above defined dimensions r, c, s.t. (e, g) is as near to the center of BB as possible, and BB is within the address space of M.
3. Then, the client generates a vector of c m-bit random numbers in Z_N^{+1}, $y = [y_1, ..., y_c]$, s.t. y_g is QNR and all other y_i $(i \neq g)$ are QR. It sends the coordinates of BB and vector y to the server.
4. The server computes for each row i of the sub-matrix BB a modular product $z_i = \Pi_{j=1}^{c} w_{i,j}$, where $w_{i,j} = y_j^2$ if $M_{i,j} = 0$, and $w_{i,j} = y_j$ if $M_{i,j} = 1$.
5. The server sends to the client $z_1, ..., z_r$.
6. The client determines that $M_{e,g} = 0$ if z_e is QR, and $M_{e,g} = 1$ if z_e is QNR.
7. Repeat step 4-6 to obtain the remaining $b - 1$ bits of the requested item in (e, g).

Fig. 2 shows the same example as in Fig. 1 but uses *bb*PIR. The general comparisons among k-Anonymity, *c*PIR and *bb*PIR in one item private retrieval are shown in Table 1. Compared to k-Anonymity, *bb*PIR is able to achieve better privacy for the same charge or a lower charge for the same privacy. The bounding box does not only include the item values that are close to the query item, but also includes item values that are not close to the query item. Compared to *c*PIR, generally if $\rho < 1/n$, $rc < n$, $c + r < 2\sqrt{n}$, the communication cost, computation cost and charge of *bb*PIR are all lower than those of *c*PIR. At one extreme, *bb*PIR degenerates into k-Anonymity if the bounding box is a column of items with close values. At the other extreme, *bb*PIR becomes *c*PIR if the bounding box is the entire matrix.

Table 1: Comparisons on One Item Private Retrieval

Method	k-Anonymity	*c*PIR	*bb*PIR
C_{comm}	$2bk$	$m\sqrt{n} + mb\sqrt{n}$	$mc + mbr$
C_{comp}	bk	mbn	$mbrc$
P_{brh}	$1/k$	$1/n$	$1/(rc)$
C_{srv}	k	$\sqrt{n}$	r

5. CONCLUSION AND FUTURE WORK

In this paper, we propose a flexible and generalized approach for private querying on public data called *Bounding-Box PIR* (*bb*PIR). This is the first paper, to our knowledge, that incorporates the realistic assumption of charging clients for exposed data.

For the future work, we consider removing the assumption that clients know the exact addresses of requested items and completely achieve querying by key. We also consider other types of private queries, such as private join of public and private data.

6. REFERENCES

[1] E. Kushilevitz and R. Ostrovsky. Replication is not needed: Single database, computationally-private information retrieval. In *FOCS*, pages 364–373, 1997.

[2] R. Sion and B. Carbunar. On the computational practicality of private information retrieval. In *Network and Distributed System Security Symposium*, 2007.

[3] L. Sweeney. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(5):557–570, 2002.

CentEx: Scalable Central Experts Search in Collaboration Networks

Petko Bogdanov
UCSB

Aydın Buluç
UCSB

Adam Lugowski
UCSB

ABSTRACT

Discovering influential researchers related to a topic is crucial for fast access to knowledge and enhanced collaboration. We present an on-line expert ranking system that emphasizes collaborations. It enables finding central researchers ranked by the betweenness centrality measure on a collaboration graph induced by a query topic. We achieve scalability through sparse data structures, efficient heuristics and multithreaded programming.

1. INTRODUCTION

We propose the problem of collaboration-aware expert search in a corpus of academic publications. The user supplies an intuitive description of a topic of interest and is presented with a ranked list of expert authors. An intuitive approach to this problem is to represent every author by a bag of terms, based solely on her papers, and perform a traditional information retrieval search. This approach would favor authors of multiple papers who are not guaranteed to be good collaborators and might not always be the experts in the field. Furthermore, if the user is interested in interdisciplinary topics, an author that "bridges" disciplines by diverse collaborations is a natural search result. To capture the notion of central collaborators, we compute a graph centrality measure for the authors in the collaboration graph compiled from the corpus. We propose ranking the authors in a query-weighted collaboration graph over the entire corpus using graph centrality metrics.

2. METHODS

In contemporary science there exists a flow of information among different scientific fields and among domains of a field. In many interdisciplinary areas there is no clear and intuitive way to categorize a paper in one specific domain. At the same time there is plurality of how scientific readers structure the space of scientific domains. Therefore, imposing a hard categorization in order to define a field of expertise is not desirable for a search engine in a scientific domain. We model authors' relationships based on papers they have co-authored. We weigh connections (papers) by their relevance to the query. In our first approach, queries and documents are probabilistically associated to latent semantic topics and we analyze the corpus using Latent Dirichlet Allocation (LDA). In our second approach, we use terms in the paper titles as representative features.

Since a publication may have multiple authors, the corpus naturally corresponds to a hypergraph $H(A, D)$ in which every author is a node and every paper is a hyper-edge connecting a set of authors. Our goal is to measure how "central" a given author is. Standard measures of centrality have been defined in the context of graphs. Hence we define a mapping from our collaboration hypergraph to a collaboration graph by replacing each hyperedge with binary edges among all co-authors. We call this mapping *collapsing* the collaboration hypergraph into a graph.

Both of our approaches use the *betweenness centrality (BC)* metric for ranking authors. The betweenness centrality of a vertex captures the ratio of the number of shortest paths that pass through a vertex to the total number of shortest paths in the graph. This is formalized below where σ_{st} denotes the number of shortest paths from s to t, and $\sigma_{st}(v)$ is the number of such paths passing through vertex v.

$$BC(v) = \sum_{s \neq v \neq t \in V} \frac{\sigma_{st}(v)}{\sigma_{st}} \quad (1)$$

We compute betweenness centrality using Brandes' [2] algorithm. It computes single source shortest paths from each node in the network and increases the respective BC score for nodes on a shortest path. The algorithm requires $O(nm)$ time for unweighted graphs and $O(nm + n^2 \log n)$ time for weighted graphs, where n is the number of nodes and m is the number of edges in the graph.

The size of the collaboration graphs is prohibitive, so we resort to efficient approximations such as *Sampling* of initial nodes and *Early Termination*. An unbiased estimator of the betweenness centrality score has been proposed based on sampling nodes [1] from which to compute single source shortest paths. In early termination, instead of completing the single source shortest paths to all nodes from a selected source node we terminate after a certain path length is reached or after a certain number of nodes have been explored. The above approximations allow for significant speedup and incur small drop in precision.

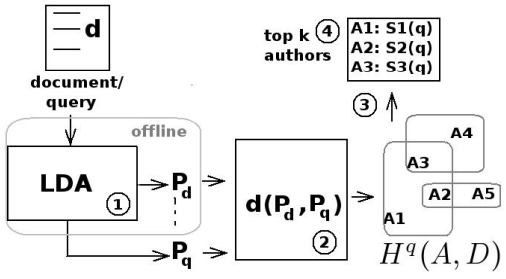


Figure 1: Overview of the LDA-based approach

An overview of the components of our LDA-based approach is presented in Figure 1. Feature vectors for all documents in the corpus are precomputed in an offline manner (phase *1* in the Figure). Given a query and documents feature vectors we compute the query specific weighted hypergraph in phase *2*. In phase *3* we collapse $H^q(A, E)$ to a graph and compute the centrality measure. Phase *4* ranks and reports the top k authors.

In our second model, both documents and queries are modeled as bags of terms and the score of a document given a query is defined as:

$$S_q(d) = \sum_{t \in q} c_d(t)/freq(t), \quad (2)$$

where $c_d(t)$ is an indicator variable that equals 1 if term t is contained in document d and 0 otherwise and $freq(t)$ is the frequency of the term in the corpus. We add the normalization by term frequency as we expect that all query terms in a multi-term query are equally important to the user. Without the normalization the centrality will always favor the central authors related to more frequent terms in the query. We build an inverted index on all terms to fasciliate scalable collapsing at query time.

The scores on author-to-author links are computed as the sum of all scores for co-authored documents. Since the BC algorithm works with distances between nodes, we convert the scores to distance using a $1/score$ mapping. Other functions can also be applied.

We use the titles, authors and venues DBLP dataset as a corpus for experimentation. It contains over 1.4 million publications and more than 700 thousand authors. We implement our method in Cilk++ (parallel multicore extension of C++) and experiment on a 16 core, 2.2Ghz Opteron, 64GB RAM system.

3. EXPERIMENTAL RESULTS

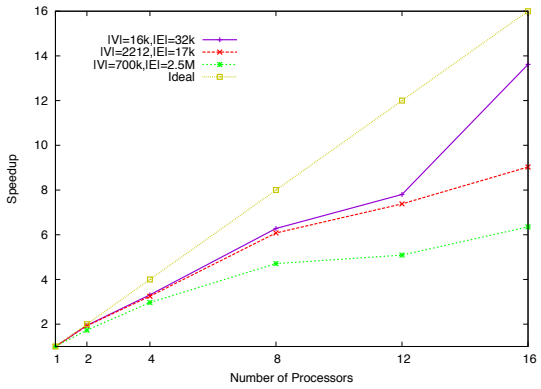


Figure 2: Speedup of our BC implementation

As Figure 2 shows, actual speedup is highly dependent on the input graph. In this experiment, the 16k node graph is an unweighted torus (4-regular graph). The 2k node graph is a gene expression graph with similar edge distribution to our collaboration graph, and the 700k node graph is a single LDA topic, sampled at 0.03%.

The search query determines the density of the search graph. We have identified several broadly worded queries which take on the order of a minute to compute the exact betweenness centrality. Since that is too long for an online system, we set a hard time limit on the BC calculation. This early stop approach is effectively sampling a subset of the nodes with the sample size adaptively determined by the length of the computation. The nodes are randomly permuted before BC is started so that no matter when the computation stops a random distribution of samples is considered.

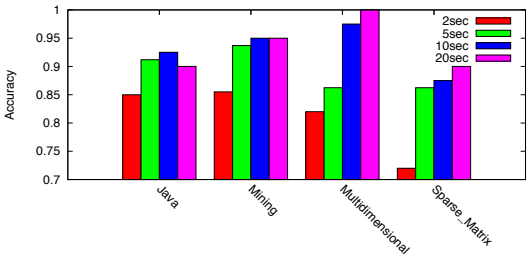


Figure 3: Accuracy with hard BC computation time limits on 4 different queries, k=20

Figure 3 shows the accuracy of sampling for a set of queries and varying time limits. Accuracy is calculated as the percentage of correctly identified top 20 authors when restricting the total query evaluation time.

We also explore the effect of sampling the full 700k node graph. Figure 4 shows that sampling just 0.03% (about 210 nodes) can yield satisfactory accuracy. Note that the gold standard is *not* the full graph but a 1% sample due to the computational complexity involved.

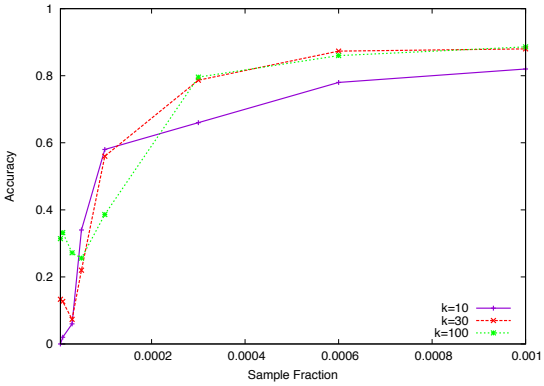


Figure 4: Sampling accuracy on 700k node, 2.5M edge LDA topic graph

4. CONCLUSION

We formulate the novel problem of searching central authors in a collaboration corpus. We compute a centrality measure on a query-centric collaboration graph in an online fashion using efficient and accurate heuristics.

5. REFERENCES

[1] David A. Bader, Shiva Kintali, Kamesh Madduri, and Milena Mihail. Approximating betweenness centrality. In *WAW*, pages 124–137, 2007.

[2] Ulrik Brandes. A faster algorithm for betweenness centrality. *Journal of Mathematical Sociology*, 25:163–177, 2001.

A Framework for Visualizing Quantum Random Walks

Arvin Faruque
Electrical and Computer Engineering Department
UC Santa Barbara
afaruque@umail.ucsb.edu

Fred Chong
Computer Science Department
UC Santa Barbara
chong@cs.ucsb.edu

Wim Van Dam
Computer Science Department
UC Santa Barbara
vandam@cs.ucsb.edu

ABSTRACT

Gaining intuition about the behavior of quantum random walks over complex graphs is difficult without the aid of computerized visualization techniques. Here, we discuss ongoing work on a scheme for interactive visualization of quantum random walks.

1. MOTIVATION

Quantum random walks are an extension of classical random walks to the quantum realm. Because of promising results about the power of algorithms based on quantum random walks ([3], [4]), gaining an intuitive understanding of the behavior of quantum random walks may lead to the development of new results in quantum information science. Barriers to gaining such an intuitive understanding of quantum random walks are the multi-dimension nature of quantum state data as well as the complex topologies of the graphs involved in the walks.

In this work, we discuss ongoing work on the design of a visualization system for interactively exploring the behavior of quantum random walks. We hope that easy access to such a system will help accelerate the development of results in this field.

2. QUANTUM RANDOM WALKS

This section provides an overview of the basic theory of quantum random walks, based off of the summary in [1].

Consider a graph G whose vertices correspond to a set of N consecutive integers $j \in \mathbb{Z}$ and whose vertices have maximum connectivity d. Corresponding to each of these vertices is a basis vector of a position space $\mathcal{H}_p$. To create a random walk, we start at an initial probability/amplitude distribution over the nodes of the graph and then proceed to randomly move to one of its neighbors in the graph. In essence, we are "flipping" a multidimensional coin at every timestep of the walk. Mathematically, we can use the notion of a "coin space" $\mathcal{H}_c$ to help us create an operator-based formalism for the random walk. We will augment the position space $\mathcal{H}_p$ with this space to attain the Hilbert space $\mathcal{H}_c \otimes \mathcal{H}_c$. Vectors in this space can then be used to store the state of the random walk at a timestep.

The process of modeling a random walk can be condensed into three steps: initial state preparation, flipping a coin and shifting amplitudes, and finally determining probabilities associated with each node. Repeating the second step yields the random walk over multiple timesteps.

1. Initial State Preparation We start with a vector $x \in \mathcal{H}_c \otimes \mathcal{H}_p$ that encodes the initial probability distribution of the location of a particle.

2. Coin Flip and Shift: To "move" our particle, we first apply a coin operator C to all the states associated with each vertex (which changes the amplitude distribution associated with each node) and then a shift operator S to shift the amplitudes across vertices.

Depending upon the requirements for our quantum walk, we can define an arbitrary unitary d-dimensional operator as our coin operator. There are also techniques for using coin operators with graphs with non-constant vertex connectivity. If we want to utilize a single coin operator of dimension d, we can add $d-m$ self loops to all vertices who who have degree $m < d$. Otherwise, it is also possible to use different coin operators for different vertices.

To define the shift operator on a graph with d-regularity, we will use the following notation, as described in [1]. Every vertex v in the graph has d edges, so we can label each of its edges with a value $j \in 1...d$. An edge e_v^j from vertex v to vertex w of the graph is the edge which is labeled j on v's end.

We can now define shift operator as the operator that moves a particle from node v to node w if an edge e_v^j exists from v to w. If we say that the state corresponding to an outgoing edge j from a vertex v is $|j\rangle \otimes |v\rangle$, then the shift operator can be summarized as follows:

$$S[|j\rangle \otimes |v\rangle] = \begin{cases} |j\rangle \otimes |w\rangle & if \quad e_v^j \; exists \\ 0 & otherwise \end{cases} \tag{1}$$

3. Probability Determination To determine the probability of being in a state j after we have modified the state vector, we sum the probabilities corresponding to each vector corresponding to each node.

3. VISUALIZATION REQUIREMENTS

Vertex Visualization Our most basic visualization requirement is to display of the vertices of the graph in its original topology along with the probabilities (and possibly the actual coefficients of the state vector) associated with each vertex. Depending on what the user is more comfortable with, this visualization can be a projection of a three-dimensional arrangement of the vertices.

Programmability When analyzing and attempting to prove results about quantum random walks, we are interested in understanding how the state of the quantum system changes after the application of different types of coin operators. This is useful for the study of algorithms that make use of a coin oracle ([5]), which can apply different coin operators to a state depending on whether a vertex is marked or not. Adding the capability to execute an arbitrary script

between every step of the simulation would facilitate this and other operations.

Collaboration If the visualization runs on a server, it is possible for one researcher to set the visualization into a certain state and add useful information (such as annotations). This in turn would make it possible for another researcher to easily view the results of the simulation.

4. PRELIMINARY RESULTS

Our work is currently in a very early stage. We have implemented a Flash applet which can simulate a classical or quantum random walk given files for an arbitrary graph structure, a coin operator, and an initial state vector. The applet injects appropriate self-loops into the graph to ensure that every vertex has uniform connectivity and performs the simulation using the same coin operator for all vertices in the graph. It can then display the output of the simulation as a string of text, a 2-dimensional plot, and as a set of colored nodes (where the color of the node corresponds to the probability of being at it at a certain timestep).

Figure 1 a) shows a screenshot of the applet after running a two steps of the walk over an 11-vertex circle graph (which is presented as an example in [1]), where there are edges between every pair of consecutive vertices. The user has the ability to drag reposition the nodes in the graph in order to better understand how the evolution is occurring. Figure 1 b) and c) show screenshots of the the plot output of the applet for a classical and quantum random walks (over the same 11-element graph after three timesteps), respectively. The initial state vector in this simulation was the state $|1\rangle \otimes |6\rangle$, and coin operators for each of these walks were the following (the classical operator uniformly maps the probabilities in the coin space, while the quantum coin is a single-qubit Hadmard gate) :

$$C_{Classical} = \frac{1}{2}\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}, \quad C_{Quantum} = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \qquad (2)$$

5. CONCLUSIONS AND FURTHER WORK

We have developed a simple system for visualizing quantum random walks. First, we plan to iteratively evaluate our system by attempting to use it to prove existing theorems about quantum random walks and running user studies. The results from these evaluations will be crucial in setting the direction of our design.

We also would like to extend this preliminary system to better meet the visualization requirements in Section 3. It may be useful add a server side (implemented using a cloud computing technology such as the Google App Engine [6]) component that can store the state of a random walk at a certain state. This will make it possible to implement multiple client-side engines for performing the quantum random walks. Of particular interest is an OpenGL-based application that would facilitate the high-speed visualization of three-dimensional complex graphs in real-time. In such an application, we can utilize a GPU with a compute driver to simulate the quantum random walk in a high-speed graphics memory buffer and then directly draw the data from this buffer using a graphics driver. Additionally, developing the native application makes it easier to add features that execute an arbitrary script on the state vector at certain timesteps in the simulation.

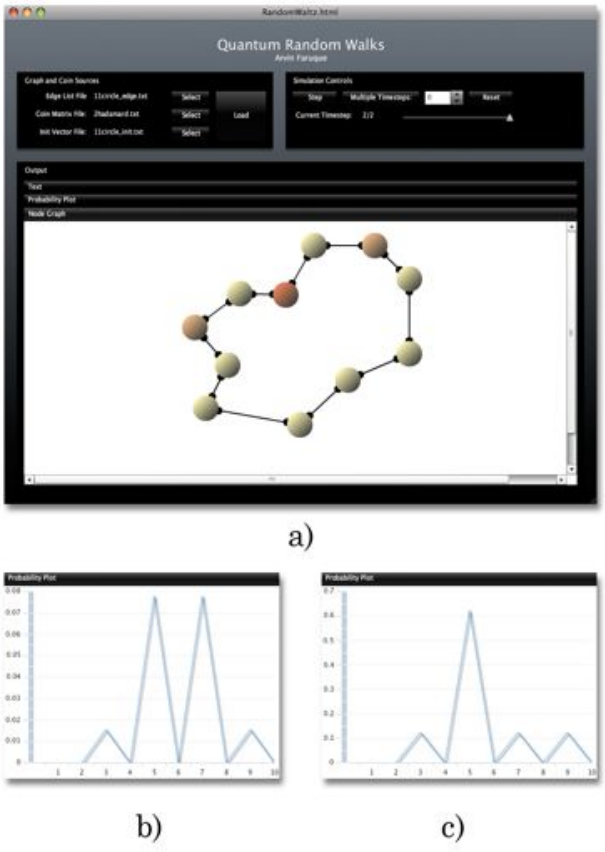


Figure 1: Flash Applet Screenshots

6. REFERENCES

[1] J. Kempe, *Quantum Random Walks: An Introductory Overview*, Contemporary Physics, 44: 307-327, 2001

[2] A. Ambainis, *Quantum walks and their algorithmic applications*, International Journal of Quantum Information, 1:507-518, 2003

[3] D. Aharonov, A. Ambainis, J. Kempe, and U. Vazirani., *Quantum walks on graphs*, In Proc. 33th STOC, pages 50:59, New York, NY, 2001

[4] A. Childs, E. Farhi, S. Gutmann., *An example of the difference between quantum and classical random walks.* Journal of Quantum Information Processing, 1:35, 2002.

[5] N. Shenvi, J. Kempe, and K.B. Whaley., *A Quantum Random Walk Search Algorithm*, Phys. Rev. A, Vol. 67 (5), 052307, 2003

[6] Google App Engine, http://code.google.com/appengine/

Evaluation of Feature Detectors and Descriptors for Visual Tracking

Steffen Gauglitz
Four Eyes Lab, UC Santa Barbara

Tobias Höllerer
Four Eyes Lab, UC Santa Barbara

1. INTRODUCTION

In recent years, there has been a lot of research on visual tracking at interactive framerates. The two most prevalent applications of visual tracking are visual navigation ("visual odometry", e.g. on NASA's Mars Exploration Rovers [1]) and augmented reality, where the user perceives live images from the real world plus computer generated elements "attached" to real world objects or locations [4]. In almost all cases, visual tracking is feature-based, i.e. first detects, describes and then matches local visual features. A variety of algorithms for these steps have been proposed and used in tracking systems, leading to an increased need for independent comparisons. Existing comparisons however focus on applications such as object recognition and image retrieval (where feature detectors and descriptors play a crucial role as well) and are of limited validity for visual tracking.

Our evaluations —in part published as technical report [2], as well as submitted to this year's ISMAR [3]—are novel in that they are specifically geared towards real-time visual tracking. In particular, we use video streams with several thousand frames naturally affected by noise and motion blur. Moreover, we evaluated the impact of several dozen algorithm parameters for both detectors and descriptors (going beyond most published work), and presented results for both stages individually as well as for any detector-descriptor combination. For this purpose, we also describe a semi-automatic method to obtain ground truth for video streams.

2. VISUAL TRACKING

Over the last years, many visual tracking systems have been proposed. They differ in motivation, aim, implementation and algorithms that are used. However, they can generally be broken down into similar main components and largely follow a structure as depicted in Figure. 1.

With a new frame captured from the video source, an interest point detector is applied to detect candidate points for tracking. For each of the candidates, a feature descriptor is computed. This descriptor is then sought to match with the descriptors of previously (e.g., in the last frame) encountered features[1]. Generally, the number of feature matches is much greater than the degrees of freedom that have to be estimated. Thus, the system is highly overdetermined,

[1]usually, the search is constrained to those of the known features that are predicted to lie close to the encountered position.

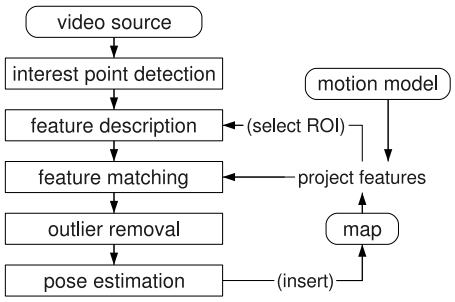


Figure 1: Generic structure of a visual tracking system.

which allows for the removal of outliers before the pose is estimated. Here, the pose could be either the pose of an object of interest or the position and orientation of the camera.

3. EVALUATION SETUP

3.1 Ground truth

To evaluate the algorithms' performance on images taken with a moving camera, ground truth information is needed, specifying which point x_j in frame j corresponds to point x_i in frame i. For general 3D scenes, this is very difficult to obtain without a 3D model of the scene. Therefore, most existing evaluations use planar or near-to-planar scenes, where x_i and x_j are related by a homography $H_{ij}(q) \in \Re^{3x3}$:

$$x_j = H_{ij}(q) \cdot x_i \tag{1}$$

(with $x_{i/j}$ in homogeneous coordinates: $x_i = (x, y, 1)^T$). Existing methods to solve for H_{ij} are not feasible for arbitrary camera motions, hence we designed a semi-automatic algorithm based on detection of small markers, adaptive color models, template matching and image alignment techniques (see [2] for details). An example of the result is shown in Figure. 2.

3.2 Testbed

Our testbed consists of 30 different video streams with 3000 frames total, showing five different planar textures in six different motion patterns each, all recorded with a resolution of 640x480 pixels. The textures were chosen to encompass features with different levels of contrast and repetitiveness of features. The motion patterns encompassed translation, rotation (both in-plane and out-of-plane), zooming, motion blur, as well as free camera movement. The camera movement is reconstructed with the method outlined in Section 3.1.

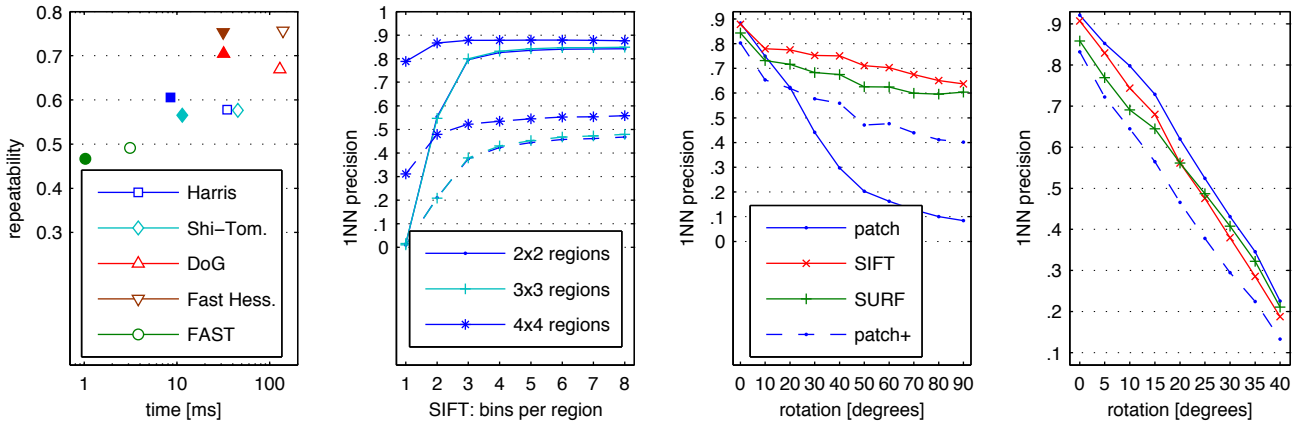


Figure 3: Exemplary results. (a) Repeatability vs. time: Repeatability (the probability that a given feature can be re-detected in the next frame) is the most important performance measure for detectors. The graph shows that the computationally more expensive detectors perform better on smooth motion, but the differences in computation time are significant (empty markers: image size 640x480, filled markers: 320x240). (b) Evaluation of parameters of the SIFT descriptor. The default settings are 4x4 regions and 8 bins per window, resulting in a descriptor vector of length 128, but it turns out that the length may be reduced significantly without decreasing performance. (c) Matching precision for different descriptors under in-plane rotation. Here, SIFT and SURF clearly outperform a simple image patch description due to the built-in orientation estimation ("patch+" is image patch combined with SURF's orientation estimation). However, in the more frequent case of out-of-plane rotation (d), the performance of all descriptors decreases quickly.

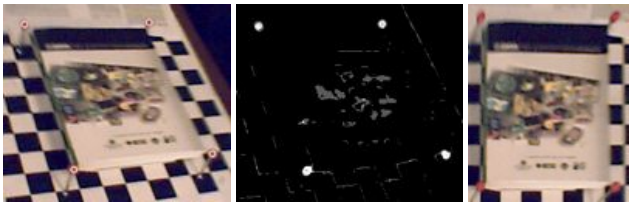


Figure 2: Solving for H_{ij}: source image (left), intermediate step illustrating color model and template matching (middle), image warped into canonical "ground truth" frame, defining H_{ij}.

The algorithms' performance was measured between consecutive frames, simulating continuous tracking during smooth motion, as well as between randomly chosen frames of a sequence, simulating tracking recovery after failure or revisiting a previously mapped scene. Therefore, every algorithm was evaluated for about 30,000 frame pairs.

4. RESULTS AND CONCLUSIONS

Given the multitude of conditions and algorithms we evaluated, full presentation of our results requires several dozen graphs. Also, as different applications may have different needs in terms of processing power, framerate, supported motions, re-usability of features etc., it is rather difficult to break the results down to a single conclusion or recommendation. We present a few examples in Figure. 3, the interested reader is referred to [2, 3] for further results.

Our results are useful in a number of ways: firstly, they show which claims about the algorithms can be confirmed in practice and which ones cannot, in particular for the image quality available in real-time tracking. They provide quantitative support for the decision of which detector/descriptor to choose for designing new tracking applications, what level of performance may be expected, and how to change parameters or algorithms if performance and/or speed have to be improved for existing applications. Furthermore, the insights gained may stimulate ideas of how existing algorithms can be improved or combined (at algorithm level or at systems level) in order to increase their performance, and which avenues are promising for more research.

5. ONGOING AND FUTURE WORK

Ongoing and future work includes extension of the evaluation to other algorithms, most notably recent machine-learning approaches that do not follow the "descriptor vector" paradigm. We are also investigating in how far our results support the development of a new detector/descriptor designed especially for visual tracking.

6. REFERENCES

[1] Y. Cheng, M. W. Maimone, and L. Matthies. Visual odometry on the mars exploration rovers - a tool to ensure accurate driving and science imaging. *IEEE Robotics & Automation Magazine*, 13(2):54–62, 2006.

[2] S. Gauglitz and T. Höllerer. In-depth evaluation of popular interest point detectors on video streams. Technical Report 2009-08, Department of Computer Science, UC Santa Barbara, May 2009.

[3] S. Gauglitz, T. Höllerer, P. Krahwinkler, and J. Roßmann. Evaluation of feature detectors and descriptors for visual tracking. *Submitted to*: the 8. IEEE and ACM Intl. Symposium on Mixed and Augmented Reality (ISMAR'09).

[4] T. Lee and T. Höllerer. Multithreaded hybrid feature tracking for markerless augmented reality. *IEEE Trans. on Visualization and Computer Graphics*, 15(3):355–368, 2009.

Expert Finding and Ranking

Matthew R. Hubert
mrh@umail.ucsb.edu

Russel J. McLoughlin
rmcl@cs.ucsb.edu

Arijit Khan
arijitkhan@cs.ucsb.edu

1. INTRODUCTION

Determining the experts for a given research area is useful for spurring collaboration, evaluating credentials of authors, and general information retrieval. As the definition of expertise can be subjective, many different factors can contribute, each having a different importance to the searcher. We set out to create a system that given a user-configurable search query, returns a list of experts using multiple criteria of expertise. Our data is based on the the DBLP dataset [4] which contains over a million entries for computer science journals and publications.

Our metrics of expertise consist of publication history, co-authorship relationships, and conference participation. Previous work in this topic such as FacetedDBLP [3] and CompleteSearch [2] take into account publication history and have strong keyword analysis but ignore inter-author relationships. Our work extends on previous methods by using an author's position within the social graph to measure expertise, as well as their participation in conferences. Additionally, our approach takes into account the publication title, contents, and authorship, extending on previous methods which only use titles.

Our search engine returns a ranked list of experts given semantic keywords. These rankings (described below) are derived by extracting and ranking keywords from each publication based on their relevance. The extracted publication keywords and relevance scores are then attributed to specific authors taking into account their contribution (authorship) to the publication (first author, second author, et cetera). Combining these relevance scores with the conference participation scores and co-author relationship scores yields a ranked list of experts in a given field.

2. RANKING ALGORITHM

The goal of our algorithm is to produce rows of author score tuples, with an author a's final score a_s represented as:

$$a_s = W_p * \text{score}_p(a) + W_c * \text{score}_c(a) + W_r * \text{score}_r(a)$$

With p, c, and r being publications, conferences, and co-author relationships, respectively. By default, $W_p = 1.0$, $W_c = 0.1$, and $W_r = 0.1$, although future work will determine these weights using a machine learning approach (3).

We define an author as a set of publications and relationships with other publications. A keyword is defined as a number of terms that embodies an atomic semantic meaning.

2.1 Publications

Given the definition of an author, we extend that definition to the publications of which said author is comprised: a publication p can be defined as a collection of keywords k_0 to k_n appearing in its title and body.

After parsing the keywords for each publication we build two indexes using Apache's Lucene, each for the publication titles and bodies. This allows us to take advantage of Lucene's built-in scoring method [1] to rank publications based on their keyword content. Lucene uses a Vector Space and Boolean model to rank its results, comparing the keywords of one publication against those of the others while limiting those publications that need to be scored in the first place.

After generating each publication score from a given user query, each author a is attributed their respective score s with n as the total number of relevant publications:

$$\text{score}_p(a) = \sum_{i=0}^{n} \frac{\text{score}_{\text{title}}(p_i) + \text{score}_{\text{body}}(p_i)}{\text{authorship}(a, p_i)}$$

2.2 Conference Participation

Intuitively, a conference can be thought of as meeting of experts on a particular topic. With this in mind it follows that the topics discussed in the conference are related to participants' areas of expertise. Therefore, to capture this relationship we define a conference as an aggregation of the keywords from each publication in said conference. Participation in a conference is defined as authoring a publication that appears in the conference's proceedings.

To calculate conference participation scores, we calculate each keyword's score and propagate it to every author using Lucene. Because a conference is comprised of such a vast quantity of keywords, we must use a separate index of conferences in order to not interfere with the regular publication scoring. Using the generated list of conference score pairs from Lucene, we propagate said conference scores for a conference c to every participating author a. With n as the number of conferences:

$$\text{score}_c(a) = \sum_{i=0}^{n} \text{score}(c_i) \text{ if } a \in \text{participants}(c)$$

2.3 Co-author Relationships

Our approach is unique in that it takes into account the relationships between authors. After every author has a score for his or her own publications, we attribute each of

those scores to those with whom they have collaborated, limited to one edge in the graph. Our rationale is based on the assumption that authors who collaborate frequently may share similar expertise.

A co-authorship score must first satisfy the following two constraints: First, the author who is receiving a score must not be an author of the publication from which they are receiving it. That is to say, an author a receives a score from all of a co-author b's other publications (which a did not write), because a has already received a score from those which he did write. Second, both publications must share a common keyword. More formally, we define this constraint as: Given A_p as the set of authors in a publication p and K_p as the set of keywords in p, we define a co-author relationship as an author a_0 who has written a publication p_0 with another author a_1, who has in turn written a publication p_1 that satisfies:

1. $a_0 \notin A_{p_1}$
2. $\exists k \in K_{p_1} : k \in K_{p_0}$

Given this constraint, an author's co-authorship score is defined below. n is the amount of publications authored by a. m is the amount of co-authors in p_i.

$$\text{score}_r(a) = \sum_{i=0}^{n} \sum_{j=0}^{m} \text{score}_p(\text{co-authors}(p_i)_j)$$

The score attribution happens for every publication a and b have authored together, so if a and b share five publications they will receive each other's scores five times. This is designed to favor authors who have collaborated multiple times, although future work will include a decay of the scores attributed after the 0th time. Note that co-authorship scores, like all scores, are weighted (mentioned under section 2.0) and thus an author will naturally not be receiving 100% of their co-author's publications.

3. EVALUATION

Our initial results presented here have limited evaluation, however current work has been focusing on extensive evaluation methods. We have developed an initial set of ground truths for certain queries—lists that represent what the correct ordering should be—using only standard publication scoring (2.1) along with Google Scholar citation information and user feedback. We train our complete method using a machine learning algorithm in order to evaluate both the accuracy of our results and the effectiveness of the individual components of our scoring (co-authorships and conferences, specifically). Additionally, future work will include additional comparisons against other DBLP search engines with improved comparison metrics.

4. RESULTS

We have tested our search engine on a variety of keywords, with each query returning in less than 3 seconds. These results only include publication titles, however future work will include the body as well. Figure 1 shows the results from a simple query, "data mining". With an execution time of 1.01 seconds on a standard Intel Core 2 machine, it returned 37328 results ranked descending by score.

Philip S. Yu	271.52972
Jiawei Han	270.81747
Jian Pei	192.77605
Wei Wang	146.00188
Xifeng Yan	136.97417
Charu C. Aggarwal	132.48347
Haixun Wang	119.50347
Jiong Yang	95.02448
Srinivasan Parthasarathy	85.68769
Jianyong Wang	85.00365

Figure 1: Top ten results of the query "data mining"

Jiawei Han	146
Philip S. Yu	130
Shusaku Tsumoto	82
Christos Faloutsos	79
Wei Wang	68
Hans-Peter Kriegel	65
Eamonn J. Keogh	63
Srinivasan Parthasarathy	63
Mohammed Javeed Zaki	61
Xindong Wu	60

Figure 2: Top ten results from FacetedDBLP

Compared with the top ten results from FacetedDBLP (Figure 2) and CompleteSearch for the same query, our results prove quite similar, with 50% being shared between at least one. Additionally, with just publication data, our algorithm ranks Jaiwei Han and Philip S. Yu first and second, respectively—the same as FacetedDBLP. However, when adding conference and co-authorship data (unlike FacetedDBLP), we see this order changing.

Unfortunately our results do suffer from some of the same problems that other search engines face with transliterated names [5] (such as Wei Wang, who indeed shows up in our top ten results), however future work could involve using a similarity metric to recognize disjoint research specialties among individuals with the same names (and thus counting them as two individuals). We also hope to improve our results further by incorporating the publication corpi in our document index, allowing authors to represent themselves among a broader range of keywords. Our search engine prototype is available at: http://ganis.cs.ucsb.edu/∼dblp/

5. REFERENCES

[1] The Apache Project. *Lucene Scoring Algorithm.*

[2] H. Bast and I. Weber. The CompleteSearch Engine: Interactive, Efficient, and Towards IR&DB Integration. In *CIDR'07 Conference Proceedings*, pages 88–95. CIDR, January 2007.

[3] J. Diederich and W. T. Balke. FacetedDBLP - Navigational Access for Digital Libraries. *Bulletin of IEEE Technical Committee on Digital Libraries*, 4(1), Spring 2008.

[4] M. Ley (Maintainer). DBLP. http://www.informatik.uni-trier.de/∼ley/db/.

[5] G. D. Sprouse. Which Wei Wang? http://pra.aps.org/PhysRevLett.99.230001.

Parker: Storing the Social Graph

Jonathan Kupferman
Dept. of Computer Science
University of California, Santa Barbara
Santa Barbara, California 93106
jkupferman@cs.ucsb.edu

Kurt Kiefer
Dept. of Electrical and Computer Engineering
University of California, Santa Barbara
Santa Barbara, California 93106
kekiefer@gmail.com

1. INTRODUCTION

Online social networks have become immensely popular in recent years. In 2008, Facebook, YouTube, and MySpace were three of the five most visited web sites in the United States. It is estimated that in early 2009, Facebook received 80 billion unique page views per month, an average of over 30,000 per second. Even if only a fraction of page views involve a data store query, handling such a large number of requests requires a high performance storage system.

Social networks may not be the only websites dealing with large amounts of traffic, but they are unique in the type of data that they host. Social networks are generally thought of as graph structures where users are nodes and relationships are the edges between them, hence "the social graph." While a useful model, graphs are known to be difficult to store and query in an efficient manner. Furthermore, many websites enforce stringent response time requirements which means latency must be kept to a minimum.

Fortunately, this data model has a few exploitable properties. First, individual node (user) lookups make up the bulk of queries. Second, the data model is straightforward. Users maintain profiles with some personal information as well as a set of friends. Third, users may update their profile or set of friends, but those updates do not need to be immediately visible to all other users. These properties allow social networking data to fit into a non-traditional database model, namely an object-oriented key-value store. A user's data can then be efficiently stored and retrieved as an object using a unique identifier as a key. Furthermore, the relaxed consistency model can allow better performance to be attained.

This paper presents Parker, a highly scalable, highly available, persistent storage system designed for storing the social graph. Parker also achieves high performance with worst-case O(1) lookup time.

2. REQUIREMENTS

API The system should provide a simple API that allows users (application developers) to perform many of the common operations required by online social networks.

Data model Since social networks vary widely in the data stored per user, it is important that developers can easily modify the data model as necessary. Versioning allows modifications to be made to the data model without requiring the entire system to be redeployed.

Performance The system should be able to handle hundreds of requests per second with minimal latency in a data center environment.

Scalability As the amount of data or requests increases, additional nodes can be added to the system in order to maintain (if not surpass) its current performance. Similarly, as any piece of the system becomes a bottleneck, the addition of servers with the same role should alleviate the issue.

3. RELATED WORK

The work presented here stems from a large body of research done by both the distributed system and database communities. Peer-to-peer systems like Chord[5] and Tapestry[6] provide intelligent mechanisms for doing look-ups in large-scale distributed systems. As a result of the high churn and failure rates commonly found in P2P systems, performing a look-up requires O(log n) hops. While these systems could be used to store graphs in a key-value manner similar that of Parker, a significant overhead is incurred as a result of the number of hops required to locate data. The additional hops result in increased latency which would make such a system impractical for use with real-time web applications. While Beehive[4] is able to provide average-case O(1) look-up performance, it assumes a Zipfian query distribution which has yet to be shown for social networks.

Amazon's Dynamo[3], Google's BigTable[1], and Yahoo's PNUTS[2] all provide high-performance, scalable storage systems, each with a different query model. Unlike the previously discussed P2P systems, these systems were designed for a data center environment where nodes join or depart the network very infrequently, nodes can communicate with low latency, and no malicious nodes exist. Unfortunately, these systems are all proprietary and are not available for public use.

4. ARCHITECTURE

Parker is designed as a two-tier horizontally scalable system, as shown in Figure 1. The back-end consists of data servers whose responsibility it is to access and modify a local database. At the front-end resides a layer of key servers. It is the task of the key servers to maintain the meta-information about the system, such as which data servers contain which pieces of data. In the simplest layout, the system has a single key server which routes all user requests to any of three replicated data servers, each running a fast, local key-value store.

The system is designed to act transparently as a centralized, shared data storage resource for social networking data. It is

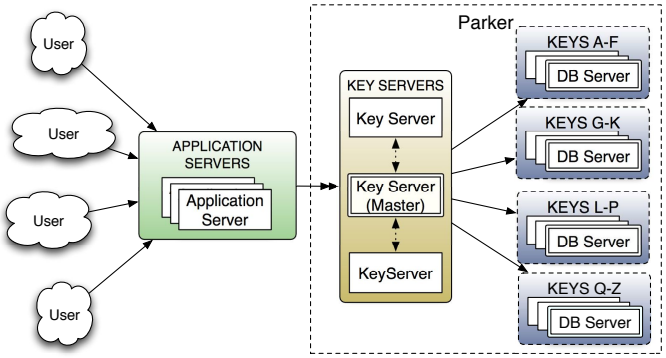


Figure 1: Parker System Architecture

primarily created for use with web applications, where multiple application servers simultaneously access shared data to generate page content for users. While this is a common case, Parker can also be used in other contexts where one needs high performance access to a very large graph structure.

Users can access social graph data via a cross-platform API that implements queries specialized for social networking applications. The provided API can be accessed via C++, Java, Python, Ruby, and Perl, among many other languages. The interface can also be easily extended to provide additional functionality beyond the original implementation – a common scenario as an application's requirements evolve over time.

5. EXPERIMENTAL RESULTS

In order to evaluate Parkers performance, two experiments were performed which explore different system properties. The first experiment was designed to evaluate the latency of the different API calls. With a key server and three data servers on machines connected via 100MBps LAN, the average latency is measured for each method. In order to measure the RPC overhead, a heartbeat method is included which is routed by a key server to a data server and immediately returns. The results of this experiment (see Table 1) demonstrate that reads are performed in a fraction of a millisecond, with a bulk of the time spent on RPC. Write calls on the other hand take significantly longer as result of replication. To ensure that data persists, each write must be received by a master data server, written to the physical medium, and then acknowledged by at least one other data server in order to succeed. This communication and I/O overhead represent most of the time spent performing write operations.

The next experiment performed was to determine the number of requests the system can handle and how well it scales with the addition of servers. Three clients were used to generate requests for three key servers and between three to nine data servers. Table 2 shows that as the number of data servers increases, a super-linear increase is achieved for writes. Since write operations are I/O bound, the additional disks are able to greatly improve performance. In contrast, read requests are unable to attain linear speed-up. This is a result of key servers becoming the bottleneck, as evidenced by the very mild improvements as the number of data servers is increased. With additional key servers this bottleneck should be be alleviated and linear speed-up should be be achieved.

Method	Response Time(ms)
heart_beat	0.2451688
get_user	0.5082938
get_profile	0.3084188
get_friends_list	0.3338142
put_user	14.14775
put_profile	18.46594
add_friend	17.89232

Table 1: Latency of API Calls

Data Servers	Reads/sec	Writes/sec
3	8067	280
6	11105	789
9	12207	1163

Table 2: Request rates on read and write workloads

6. CONCLUSION

We have presented a novel architecture for storing social network data in a distributed environment. While previous systems have been created to store graph information, we have shown that storing graph nodes as key-value pairs allows the system to have a natural design while achieving high performance in a real-time environment. An additional benefit is that such an architecture is easily scalable to various types of load. It can also be easily extended to meet the requirements of most online social networks. Thus we believe Parker is suitable for use in building the next great online social network.

7. REFERENCES

[1] F. Chang, J. Dean, S. Ghemawat, W. Hsieh, D. Wallach, M. Burrows, T. Chandra, A. Fikes, and R. Gruber. Bigtable: A distributed storage system for structured data. In *Proceedings of the 7th USENIX Symposium on Operating Systems Design and Implementation (OSDI'06)*, 2006.

[2] B. Cooper, R. Ramakrishnan, U. Srivastava, A. Silberstein, P. Bohannon, H. Jacobsen, N. Puz, D. Weaver, and R. Yerneni. PNUTS: Yahoo!'s hosted data serving platform. *Proceedings of the VLDB Endowment archive*, 1(2):1277–1288, 2008.

[3] G. DeCandia, D. Hastorun, M. Jampani, G. Kakulapati, A. Lakshman, A. Pilchin, S. Sivasubramanian, P. Vosshall, and W. Vogels. Dynamo: amazon's highly available key-value store. *ACM SIGOPS Operating Systems Review*, 41(6):205–220, 2007.

[4] V. Ramasubramanian and E. Sirer. Beehive: O (1) lookup performance for power-law query distributions in peer-to-peer overlays. In *Symposium on Networked Systems Design and Implementation (NSDI'04)*.

[5] I. Stoica, R. Morris, D. Karger, M. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 149–160. ACM New York, NY, USA, 2001.

[6] B. Zhao, J. Kubiatowicz, and A. Joseph. Tapestry: An infrastructure for fault-tolerant wide-area location and routing. *Computer*, 74, 2001.

A Replication Study Testing the Validity of AR Simulation in VR for Controlled Experiments

Cha Lee*
University of California, Santa Barbara

Scott Bonebrake†
University of California, Santa Barbara

Tobias Höllerer‡
University of California, Santa Barbara

Doug A. Bowman§
Virginia Tech

ABSTRACT

It is extremely challenging to run controlled studies comparing multiple Augmented Reality (AR) systems. We use an "AR simulation" approach, in which a Virtual Reality (VR) system is used to simulate multiple AR systems. In order to validate this approach, we carefully replicated a well-known study by Ellis et al. using our simulator, obtaining comparable results.

Index Terms: I.3.7 [Three-Dimensional Graphics and Realism]: Virtual Reality—AR Simulation; I.3.6 [Methodology and Techniques]: Device independence—Replication

1 INTRODUCTION

The inherent components of AR pose challenging issues when trying to conduct controlled experiments between systems. Hardware for AR systems can differ greatly and finding compatible systems may prove to be extremely hard. AR applications also typically rely on the real world as a backdrop to virtual objects. This makes replicating the exact real world conditions impossible in outdoor applications and difficult for indoor applications. We use an "AR simulation" approach, in which a VR system is used to simulate a range of AR systems. AR simulation, as proposed by Gabbard et al. [2] and Ragan et al. [3], may prove to be a viable method of conducting controlled experiments in AR. By using VR, we can simulate real world conditions and hardware configurations (display, tracking, processing power) of an AR system and provide repeatable conditions for experiments. But are the results of experiments using AR simulation valid for real-world AR systems? Our research begins to address this question by attempting to replicate a well-known AR experiment.

2 DESIGN

The experiment we chose to replicate was the second study in Ellis et al. [1]. Our VR system consisted of a Kaiser Proview 60 HMD, WorldViz Precision Position Tracking (PPT) system, two wired Intersense InertiaCube2, and a VRML model based on the ReCVEB lab at UCSB. To simulate the original HMD used, we rendered a transparent region equivalent to the 19 deg field of view of the original HMD. All augmentations were only viewable within this region while the simulated real objects could be seen in the entire view of the user, as shown in Figure 1. The two Intersense InertiaCube2 in conjunction with three markers (two for head, one for hand) from the PPT system were used to track the head and hand of the user in 6 degrees of freedom. A *simulated* real hand was then used to represent the actual hand of the participant.

*e-mail: chalee21@cs.ucsb.edu
†e-mail: egscott2004@gmail.com
‡e-mail: holl@cs.ucsb.edu
§e-mail: bowman@vt.edu

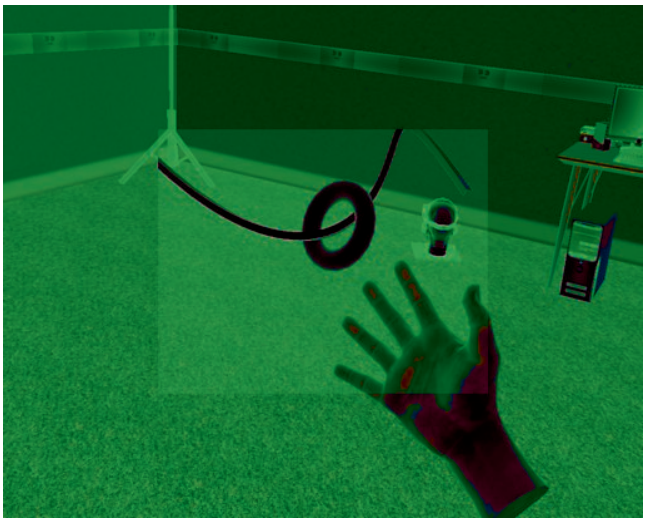


Figure 1: Screenshot of participant's view

For the experiment we had 14 users, ages 21 - 33, nine male and five female. All users were comfortable around computers. All users were able to perceive stereo as verified by a random dot stereogram. The HMD was then automatically calibrated and registered with each user with a constant error of less than two cm. Previous work [4] has found evidence that users are able to adapt to small perturbations in head registration for motor performance in VR tasks. Participants were asked to trace a virtual 3D path with a virtual ring rigidly attached to the participant's hand. Independent variables were ring size, path type (angular and smooth), and total system latency (on the virtual objects). There were five different latencies and six different paths for a total of 30 different conditions per ring type. The conditions were randomly ordered in a block of trials and each participant performed three blocks. Seven participants used the small ring, and seven the large ring. The dependent measure in this experiment was the number of collisions between the ring and the path. For more details on the experimental design, please refer to [1].

3 RESULTS

Preliminary analysis of the results showed that users reached asymptotic performance after the first block of trials; thus only the second and third blocks were used in the following analysis. All significant effects on tracing performance are shown in Table 1. These results are comparable to the results from [1] with our results also showing a significant interaction between path and latency.

In addition to comparable statistically significant effects, all of the effects were in the same direction as in the original study: the number of collisions increased with angular paths, with the small

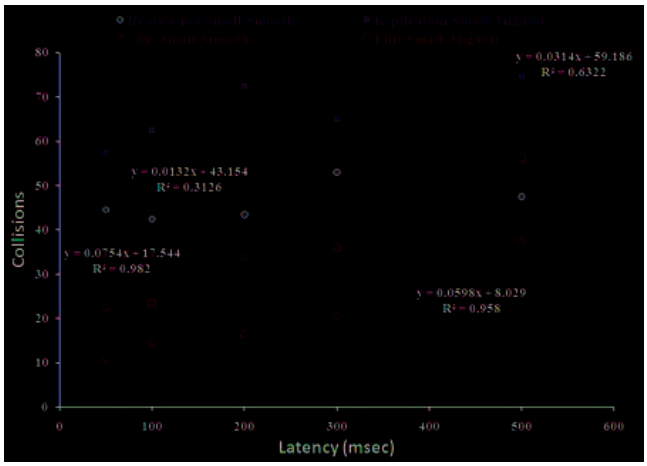


Figure 2: Comparison of trend lines for tracing performance of the small ring. Our replication study results are represented by the solid blue data points.

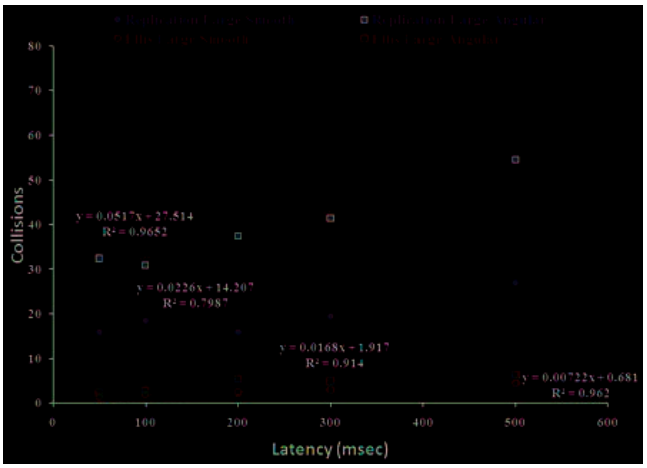


Figure 3: Comparison of trend lines for tracing performance of the large ring. Our replication study results are represented by the solid blue data points.

ring size, and as latency increased. However, there were some interesting differences in the absolute performance data, as shown in Figures 2 and 3. The participants' absolute tracing performance is much worse in our experiment. We believe this is mostly due to the differences in the collision algorithms used. It was unclear how the original experiment determined collisions during sustained contact of the ring and path. Initially we only counted this type of collision once, but this benefited the careless participants in the pilot study we ran. As a result, we decided to add collisions for every 200 msecs the ring stayed in contact with the path which increased collisions overall but penalized the careless participants. Ellis et al.'s experiment also showed a visible increase in collisions as system latency increased for the small ring. This effect was less observable and in fact starts to level off as if a ceiling effect was occurring in our experiment. Due to the lower clearance for the small ring, users moved their head extremely close to their hand to get a better view which occasionally caused small jitter issues, because PPT is a vision tracking system which depends on line of sight. Adding our rigorous collision criteria, the small ring task may have been too difficult within our system.

4 Discussion on Experimental Replication

During the course of this work, we learned some valuable lessons with regards to replicating and simulating previous experiments. One important reason for choosing the experiment in [1] was the very detailed description of the design and analysis the authors provided in the paper. In addition, Dr. Ellis was kind enough to be available for questions. Even with all this, understanding the original setup and design of the experiment was extremely challenging. We did not have access to the original models used, lacked information on the environment except for the video provided, lacked access to the original collision algorithm, and lacked the raw results from the previous study. Replicating the original models was impossible, so our models were only approximations. In summary, it would be extremely hard if not impossible to repeat any experiment without very detailed notes or the guidance of the original authors. This highlights the importance and need for even more detailed reports on experiments in the community. Although this may not be feasible within a conference or journal paper format, this information is invaluable to the repeatability of these experiments.

Table 1: Significant Effects on Tracing Performance

Effect	df	F level	
Ring	1, 12	9.075	$P < 0.011$
Path	1, 12	25.638	$P < 0.001$
Latency	4, 48	14.245	$P < 0.001$
Path x Latency	4, 48	7.484	$P < 0.001$
Path x Ring x Latency	4, 48	3.348	$P < 0.017$

5 Conclusion and Future Work

We have repeated a well known AR experiment by Ellis and colleagues and obtained comparable results. We believe this is a step toward validation of the idea that AR simulation is a viable and effective method of performing controlled experiments. For future work, we are interested in experimenting with other factors in AR systems such as display size, field of view, and resolution within our simulation environment.

Acknowledgements

We would like to thank Dr. Stephen R. Ellis for his time and invaluable guidance in replicating the original study. We would also like to acknowledge Masaki Miyanohara and the ReCVEB lab at UCSB for providing the virtual model of their lab.

References

[1] S. R. Ellis, F. Breant, B. Manges, R. Jacoby, and B. D. Adelstein. Factors influencing operator interaction with virtual objects viewed via head-mounted see-through displays: viewing conditions and rendering latency. In *VRAIS '97: Proceedings of the 1997 Virtual Reality Annual International Symposium (VRAIS '97)*, page 138, Washington, DC, USA, 1997. IEEE Computer Society.

[2] J. L. Gabbard, J. E. Swan, II, and D. Hix. The effects of text drawing styles, background textures, and natural lighting on text legibility in outdoor augmented reality. *Presence: Teleoper. Virtual Environ.*, 15(1):16–32, 2006.

[3] E. Ragan, C. Wilkes, D. A. Bowman, and T. Höllerer. Simulation of augmented reality systems in purely virtual environments. *Virtual Reality Conference, IEEE*, 0:287–288, 2009.

[4] D. W. Sprague, B. A. Po, and K. S. Booth. The importance of accurate vr head registration on skilled motor performance. In *GI '06: Proceedings of Graphics Interface 2006*, pages 131–137, Toronto, Ont., Canada, Canada, 2006. Canadian Information Processing Society.

Soft Coherence: Preliminary Experiments with Error-Tolerant Cache Coherence in Numerical Applications

Guoping Long†, Frederic T. Chong‡, Diana Franklin‡, John Gilbert‡, Dongrui Fan†
† Institute of Computing Technology, Chinese Academy of Sciences
‡ Department of Computer Science, UC Santa Barbara

1 Introduction

The ever increasing gap between the speed of the processor and memory, also known as the memory wall problem, has drawn extensive attention from the research community. The value prediction [1] and silent stores [2] seek to mitigate the overheads of memory loads and stores, respectively. Another interesting work is coherence decoupling [3], which allows loads to return values prematurely, and relies on hardware recovery mechanisms to ensure correct protocol operation. However, all these previous works have two fundamental limitations. First, to ensure correctness, each load can only obtain the value produced by the most recent store. Second, all these optimization techniques rely on aggressive speculation to achieve performance gain, which inevitably introduces notable extra hardware complexity.

In this work, we explore a third dimension of design space beyond performance and hardware complexity, the application fidelity. The key observation is that many applications allow a certain degree of algorithmic error resilience. For example, given a parallel numerical application, if the required precision (the difference between the execution output and the theoretical result) is $1e-10$, and strict implementation of cache coherence can achieve the precision of $1e-15$, then there is room for relaxation of operations which are not critical to the output precision. We leverage this property to design *soft coherence* protocols. Our design eliminates the hardware overhead for mis-speculation recovery completely by allowing some loads to obtains stale values.

We explore analysis methods to distinguish between critical and non-critical data in such algorithms, and provide strong guarantees for critical data and weak guarantees for non-critical data. We evaluate the potential of soft coherence protocols with a conjugate gradient (CG) solver on the Godson-T many core platform [4]. Section 2 presents the technique to identify the critical data for CG solver, and the hardware support for soft coherence. Section 3 discusses some preliminary results and future work.

2 Soft Coherence

2.1 The Parallel CG Solver

The conjugate gradient (CG) solver here is adapted from the CG program of the NPB benchmark suite. This program estimates the largest eigenvalue of a symmetric positive definite sparse matrix with the inverse power method. We refer readers to [5] for basic parallelization strategy for the kernel loop. In addition to this basic parallelization, we partition the sparse input matrix with Mondriaan [6] to achieve better load balancing.

For soft coherence, we are interested in the changing of sharing patterns of those shared vectors during the computation. Specifically, there are two types of operations in the kernel loop. One is the canonical vector operations. The other one is the sparse matrix to vector multiplication. Since we partition the sparse matrix for load balancing, as a side effect, this also introduces an irregular access pattern to the shared vector to be multiplied. The variation of sharing pattern on shared vectors incurs coherence traffic among processors. The goal is therefore to identify the set of memory op-

erations which cause coherent misses, and relaxing the semantics of these operations has little or no effect on the output precision and convergence time.

2.2 Critical Data Identification

For each input variable x, we can view the numerical solver as a partial function of x:$f(x)$. Now assume x changes from a to b. The problem is how to evaluate the impact of this input variation on the output: $f(b) - f(a)$. Since $f(x)$ is usually complex, we turn to its Taylor series with respect to x. When all input values are given, the Taylor coefficients can be obtained with automatic differentiation tools, such as Rapsodia [7]. In this work, we use coefficients from the first order to the fifth order to approximate $f(b) - f(a)$.

Even if we know the relative importance of each input memory location, the central question is to decide which operations to relax at each iteration of the kernel loop. In this paper, we call this relaxation plans (RP). Obviously, different inputs require different RP. The basic rational for RP generation are two fold. First, to ensure the least loss of output precision, we choose to relax those operations with least criticalness. We need an input dependant threshold to control the number of operations to relax at each iteration. Second, to ensure convergence, we never relax any operations in two consecutive iterations.

2.3 How Soft Coherence Works

Figure 1 shows how soft coherence works as a whole system. The input sparse matrix serves as the input to both Rapsodia and the CG solver. Rapsodia generates coefficients of Taylor series, which are used to determine the relaxation plan for the input. We design soft coherence hardware support on Godson-T to exploit the hints provided by RP, and execute multi-threaded CG solver with selective relaxation of operations on non-critical memory locations.

3 Preliminary Results and Future Work

We evaluate the potential of soft coherence on Godson-T many core simulator, which differs in a few important ways (in-order cores, block caches, no directory protocol support, etc) to traditional

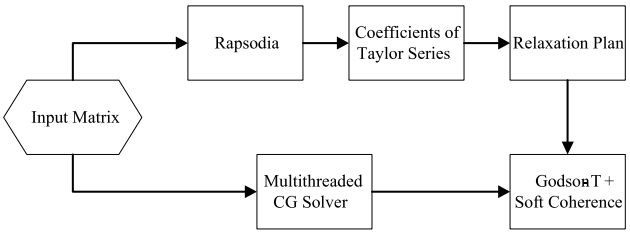


Figure 1: Overview of Soft Coherence

directory-based invalidate protocols. We design necessary hardware support for soft coherence. Experimental results on the conjugate gradient solver show that 6.9% to 12.6% performance improvement can be achieved.

In future work, we will evaluate more applications on a more general system running an invalidation-based cache coherence protocol.

References

[1] M.H.Lipasti, C.B.Wilkerson, and J.P.Shen, "Value locality and load value prediction," in *Proceedings of International Conference on Architectural Support for Programming Languages and Operating Systems*,

[2] K. M. Lepak and M. H. Lipasti, "Silent stores for free," in *Proceedings of International Symposium on Microarchitecture*, December 2000.

[3] J. Huh, J. C. Chang, D. Burger, and G. S. Sohi, "Coherence decoupling: Making use of incoherence," in *Proceedings of International Conference on Architectural Support for Programming Languages and Operating Systems*, October 2004.

[4] G. P. Long, D. R. Fan, and J. C. Zhang, "Architectural support for cilk computations on many core architectures," in *Proceedings of ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, Febrary 2009.

[5] H. Lof and J. Rantakokko, "Algorithmic optimizations of a conjugate gradient solver on shared memory architectures," *The International Journal of Parallel, Emergent and Distributed Systems, Vol. 21, No. 5, pages 345-363* , 2006.

[6] B. Vastenhouw and R. H. Bisseling, "A two-dimensional data distribution method for parallel sparse matrix-vector multiplication," *SIAM Review, Vol. 47, No. 1, page 67-95* , January 2005.

[7] I. Charpentier and J. Utke, "Fast higher-order derivative tensors with rapsodia," *Optimization Methods Software* , 2009.

Towards Real-time Prediction of Network Traffic Load

Ashish Sharma
University of California, Santa Barbara
asharma@cs.ucsb.edu

Veljko Pejovic
University of California, Santa Barbara
veljko@cs.ucsb.edu

I. Introduction

This paper discusses real-time network prediction system design from a novel perspective. We leverage the recent advancements in the field of data mining in order to solve a well established computer networking problem. Our approach is to group network flows that exhibit similar behavior into clusters and then develop a classification scheme that is able to identify the flows in real-time. The main obstacle in doing so is that it is not well understood what the most important features are in a network flow trace, from a clustering perspective. We have looked at characteristics such as inter-packet arrival times, packet sizes, and time of packet transmission of a packet since the beginning of a flow. We have received mixed results from an initial set of analysis results and the efforts are ongoing.

The applications of a network load prediction system are manifold. Fast and effective prediction technique to estimate the traffic requirement of a connected client can help IEEE 802.16e (WiMAX) based access points to assign sub-channels comprised of a set of sub-carrier frequencies, for a certain number of time slots as need; in a system with the electronically steerable antennae technology, the access point could focus the signal towards a client depending on the traffic load between the client and the access point; and finally in a network of self powered wireless routers the quality of service that they provide is bounded by the availability of energy supplies. Shutting down unnecessary network nodes when not in use extends network uptime and positively impacts the performance. Without accurate information about the future traffic load it is impossible to power the nodes back on time and at the same time optimize energy savings.

II. Clustering Network Flows

A network flow comprises of data packets exchanged between communicating hosts on a network using transport layer protocols. The inter-arrival time between two consecutive packets of a network flow transmitted by a host is a complex function determined by the application traffic generation rate, the transport layer protocol in use, queuing delays at the host and on the intermediate nodes in the network, the medium access protocol, and finally a random amount of jitter.

Wireless Network Traces: In our experiments, we use the wireless network traces collected at the Dartmouth college campus available at the open CRAWDAD network trace repository. The network traces are available as unencrypted captures of packet headers including the IP and transport layer headers but not the data. While the actual data packet may be up to 1500 bytes in size, the trace consists of only the 50 first bytes of a packet. We analyze almost 400 MBs of packet header data, which comprised of roughly 800,000 flows and 4.5 million packets. We define a network flow as a (src-ip, src-port, dst-ip, dst-port) tuple. The data set also contains application labels for each flow based on the port used by the transport protocol.

A. Packet sequences of inter-arrival times

In this representation, a traffic flow is seen as a sequential array of packet interarrival times. The algorithm for exact time series matching[1] is proposed for normalized time-series. The algorithm is appealing as the resulting similarity matrix allows classification even in case of small flow variations that cannot be captured if the flows are compared on a simple packet-per-packet basis. The algorithm was proposed to match time-series that contain the same number of observations on a uniform y-axis. Unfortunately, the approach is not suitable in cases where the flows drastically differ in their length, or when there is a significant time offset between characteristic packet patterns.

We perform the clustering on a mix of TCP-DAAP and TCP-DOCENT flows, as the two types are predominant in our trace. The resultant dendrogram reveals that the flows corresponding to TCP-DAAP protocol appear as multiple small clusters, while the TCP-DOCENT flows show little similarity. This is expected, since a closer look at the flows shows that TCP-DOCENT flows have a high degree of randomness. Based on our clustering result, and the analysis we skip here for brevity, we make the following two observations:

1) Time-series matching is effective for short flows.
2) For longer flows, hierarchical clustering performs better when the flow is divided in different short phases.

B. Histogram of inter-arrival times

While the first approach of time-series matching is able to capture differences in short flows, for longer flows it becomes susceptible to noise and hence inaccurate classification. We now explore a second approach of transforming a flow in to a histogram representation.

[1] A. Mueen, E. Keogh, Q. Zhu, S. Cash, and B. Westover, "Exact Discovery of Time Series Motifs". In SIAM, Sparks, NV, April 2009

At the first thought, the interarrival time histogram is not a good way to represent a time series: one histogram can correspond to a number of different time series. However, in our problem domain the time series are not generated randomly. They are defined by network protocols. Thus, the histogram representation has a high degree of correspondence with one or a small number of time series behaviors. To evaluate this hypothesis we visually inspected histograms and time series of about a dozen traces generated by more or less known applications. Figure 1 shows one such plot: similar time series have similar histograms while the differing time series have differing histograms. Encouraged by the results,

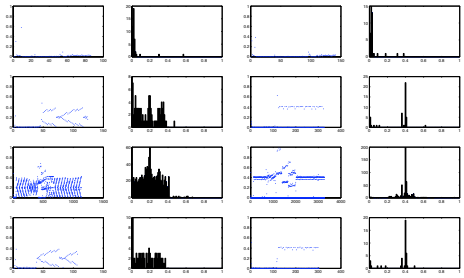


Fig. 1: Time Series and Histogram representation of 8 sample TCP - DAAP flows

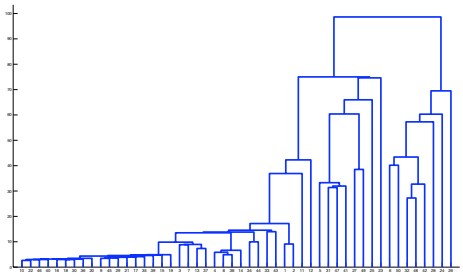


Fig. 2: Dendrogram of fifty TCP - DAAP histogram representations

we decided to use the interarrival time histograms for flow classification. We measure the inter-arrival times of packets and place them in bins of 10ms each and create a histogram of the observed flow. We create 100 bins of 10ms each [0-10, 10-20, 20-30, ... 990-]. The comparison between each pair of histograms is performed and L2 (Euclidean distance) is used. The similarity matrix is then fed to the hierarchical clustering method and the dendrogram is created. We tested the clustering process on flows selected from the Dartmouth campus trace. A sample dendrogram is plotted in Figure 2 and shows the hierarchical clustering result for the histogram representation of 50 TCP - DAAP flows. A large majority of these flows contain several thousand packets. Similar flows are clustered together, while those that are different are mutually distant in the dendrogram. We observe the improvement over the direct time series motif clustering. The reason for that is the histogram's insensitivity to temporal variations between flows and length of flows.

C. Bidirectional flows as packet sequences

So far, each flow was considered unidirectional with outgoing packets and incoming packets treated as two distinct flows. Such a representation does not contain the request response behavior of a bidirectional flow. For instance, TCP protocol waits for an acknowledgement for every few packets based on its current window size. To capture this latent inter-dependency in a bi-directional flow, we model the network flows as bidirectional exchange of packets. We label all outgoing packets as **A** and all incoming packets as **B**. The resulting flow representation is transformed in to a sequence of **A**s and **B**s.

Our aim is to identify any characteristic patterns such as a repeating sequence of **ABBABBABB..**. The repeating pattern of **ABB** might reveal information about the number of incoming packets (**B**) seen for every outgoing packet (**A**). The difficulty in such an approach is that the size of the recurring pattern (k) is unknown. We experiment with different pattern sizes of k=4, 5, and 6. We use a sliding window to count the number of patterns seen in a flow.

It is interesting to note that despite an unknown value of k, it is possible to determine the correct value of k in some cases. This is possible when the pattern occurs consecutively and the sliding window size is comparable to the pattern. To elaborate this point in detail, let us consider an example. If the recurring pattern **ABB** of size *k=3* occurs consecutively as **ABBABBABB..**, at several places in the flow, then a sliding window of size *k=4* would see three different patterns occurring with similar frequency, namely: **ABBA, BBAB, BABB**. In the future, we intend to associate timestamp information to the pattern occurrences to determine higher levels of inferences.

III. Conclusion

In this paper, we explore the novel direction of network data mining. We adopt the approach of representing a network flow as a time-series and cluster network flows that have similar behavior. In the future, we intend to use such a clustering, to identify the characteristic patterns defining each cluster and then using such patterns to classify and predict network flow behavior. We explore three different time-series representation of network flows and apply hierarchical clustering to these representations. Our results so far indicate that a time-series of $< packetnumber, interarrivaltime >$ tuples is best suited for flows with a smaller number of packets (approx. 50 packets). For longer flows, we look at a histogram representation that is able to cluster similar network flows together. Finally, we explore the representation of a network flow as a bidirectional transfer to detect latent inter-dependencies in request-response based network flows.

While we have explored several approaches for network flow classification, we believe this is just the beginning and in the future we would like to continue this exploration to ultimately be able to predict network flows with a high degree of accuracy.

Complete Information Flow Tracking from the Gates Up

Mohit Tiwari, Hassan M G Wassel, Bita Mazloom, Frederic T Chong and Timothy Sherwood
Department of Computer Science, University of California, Santa Barbara
{tiwari,hwassel,betamaz,shashimc,chong,sherwood}@cs.ucsb.edu

1 Summary

The enforcement of information flow policies is one of the most important aspects of modern computer security, yet is also one of the hardest to get correct in implementation. The recent explosion of work on dynamic dataflow tracking architectures has led to many clever new ways of detecting everything from general code injection attacks to cross-site scripting attacks. The basic scheme keeps track of a binary property, trusted or untrusted, for every piece of data. Data from "untrusted" sources (e.g. from the network) are marked as untrusted, and the output of an instruction is marked as untrusted if any of its inputs are untrusted. While these systems will likely prove themselves useful in a variety of real-life security scenarios, precisely capturing the flow of information in a traditional microprocessor quickly leads to an explosion of untrusted state because information is leaked practically everywhere and by everything. If you are executing an exceedingly critical piece of software, for example, using your private key to sign an important message, information about that key is leaked in some form or another by almost everything that you do with it. The time it takes to perform the authentication, the elements in the cache you displace due to your operations, the paths through the code the encryption software takes, even the paths through your code that are never taken can leak information about the key.

We present for the first time a processor architecture, analysis technique, and proof-of-concept implementation that can precisely track *all* information-flows[1]. On such a microprocessor it is impossible for an adversary to hide the flow of information through the design, whether that flow was intended by both parties (e.g. through a covert channel) or not (e.g. through a timing-channel). One of the key insights in this paper is that all information flows, whether implicit, covert, or explicit, look surprisingly similar at the *gate level* where weakly defined ISA descriptions are replaced by concrete logic functions.

1.1 Gate-Level Information Flow Tracking

Consider an AND gate (shown in left side of Figure 1) with two binary inputs, a and b, and an output o. Let's assume for right now that this is our entire system, and that the inputs to this AND gate can come from either trusted or untrusted sources, and that those inputs are marked with a bit (a_t and b_t respectively) such that a 1 indicates that the data is untrusted (or "tainted"). The basic problem of gate-level information flow tracking is to determine, given some input for a and b and their corresponding trust bits a_t and b_t, whether or not the output o is trusted (which is then added as an extra output of the function o_t).

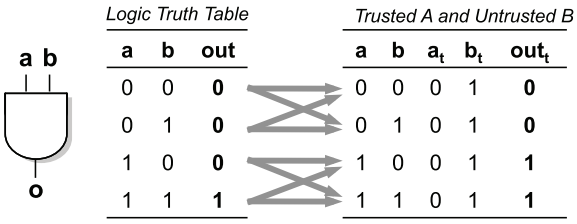


Figure 1: Tracking Information Flow through a 2-input AND Gate. Figure shows truth table for the AND Gate (left) and a part of its shadow truth table (right). The shadow truth table shows the interesting case when only one of the inputs a and b is trusted (i.e. $a_t = 0$ and $b_t = 1$). Each row of the shadow table calculates the trust value of the output (out_t) by checking whether the untrusted input b can affect the output out. This requires checking out for both values of b in the table on the left. The gray arrows indicate the rows that have to be checked for each row on the right. For example, when $a = 1$, b affects out (row 3 and 4 on the left). Hence row 3 and 4 on the right have out_t as untrusted.

The assumption that most prior work makes is that when you compute a function, any function, of two inputs, then the output should be tagged as tainted if *either* of the inputs are tainted. This assumption is certainly sound (it should never lead to a case, wherein output which should not be trusted is marked as trusted) but it is over conservative in many important cases, in particular if something is known about the actual inputs to the function at runtime. To see why, let us just consider the AND gate, and all of the possible input cases. If both of the inputs are trusted, then the output should clearly be trusted. If both the inputs are untrusted, the output is again clearly untrusted. The interesting cases are when you have a mix of trusted and untrusted data. If input a is trusted and set to 1, and input b is untrusted, the output of the AND gate is always equal to the input b, which, being untrusted, means that the output should also be untrusted. However, if input a is trusted and set to 0, and input b is untrusted, the result will always be 0 regardless of the untrusted value. The untrusted value has absolutely no effect on the output and hence the output can inherit the trust of a. By including the actual values of the inputs into the determination of whether the output is trusted or not trusted, we can more precisely determine whether the output of a logic function is trusted or not.

While this seems like an awful lot of trouble to track the information flow through an AND gate, without

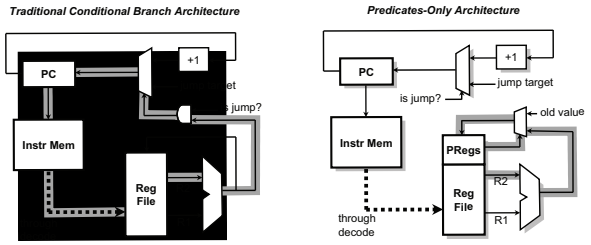


Figure 2: Implementation of a conditional branch instruction in a traditional architecture compared to ours. The highlighted wires on the left figure shows the path from an untrusted conditional to the PC. In contrast, we eliminate the path in our architecture so that the PC never gets untrusted.

this precision, there would be no way to restore a register to a trusted state once it has been marked untrusted. Its impact in terms of the ability to build a machine that effectively manages the flow of information is immense.

1.2 Overview of Results

Compositional GLIFT Technique: While the truth table method that we describe above is the most precise way of analyzing logic functions, our end goal is to create an entire processor using this technology. Enumerating the entire truth table for our full processor (which would have approximately 2^{769} rows, where 769 is the number of state bits in our processor prototype) is not feasible, therefore we need a way of composing functions from smaller functions in a way that preserves the soundness of information flow tracking. In the paper we describe how GLIFT logic is conservatively compositional, meaning that untrusted data will never be marked incorrectly as trusted. We demonstrate this with the example of a multiplexer (where the trust of the output is related to the trust of the selected input), which we then use throughout our processor implementation.

GLIFT as an Analysis Technique: After discussing the GLIFT logic method, the next question then becomes how that method can be applied to a programmable device to create an air-tight information flow tracking microprocessor. The goal of our architecture design is to create a full *implementation* that, while not terribly efficient or small, is programmable enough and precise enough in its handling of untrusted data that it is able to handle several security related tasks, while simultaneously tracking any and *all* information flows emanating from untrusted inputs. Figure 2 shows a simple example of a branch instruction implemented in hardware and the problem with traditional architecture. Once a comparison occurs on untrusted data, the result is used to control the select line to the Program Counter, which means the PC can no longer be trusted. Once the PC is untrusted, there is no going back because each PC is dependent on the result of the last. In the architecture described above, all instructions after a branch on trusted data will be marked as untrusted, but is information really flowing in that way? In fact, at the gate level, it is. There is a *timing* dependence between the value of the branch predicate and the time at which the following instructions are executed. Such timing observations, while seemingly harmless in our example, do represent real information flow and have been used to transmit secret data and reverse engineer secret keys.

A Proof-of-Concept Information-Tight Architecture: As is apparent from our previous example, traditional conditional jumps are problematic, both because they lead to variations in timing and because information is flowing through the PC (which has many unintended consequences). Predication, by transforming if–then–else blocks into explicit data dependencies (through predicate registers), provides one answer. Handling loops requires a different approach. Loops are surprisingly difficult to constrain as there are so many different ways for information to leak out in non-obvious ways. Consider a simple while-loop on an untrusted condition. Every instruction in that loop may execute an arbitrary number of times, so everything those instructions touch is untrusted. In fact, everything that *could have been modified*, even if it wasn't, needs to be marked as untrusted (due to implicit flows). In the paper we discuss solutions to if-then-else control, looping, memory references, and describe a new ISA that is verifiably presents no opportunities for information leakage.

An Full Verifiable Implementation running on an FPGA: Because GLIFT is provides a way to verify the information flow properties of a full implementation running different software, we have built a working processor and its GLIFT augmentation on an FPGA, and we have written several application kernels to help us quantify the overheads involved. We use Altera's Nios processor as a point of comparison as it has a RISC instruction set, and, as a commercial product, is reasonably well optimized. Our base processor is almost equal in area to Nios-standard, and about double the size of Nios-economy. Adding the information flow tracking logic to the base processor increases its area by 70%, to about 1700 ALUTs. Static code sizes are very similar between the different implementations. While the overheads involved are certainly non-trivial, by trading off precision for efficiency it may be possible to keep the soundness of our result while reducing the performance impact. However, our prototype demonstrates for the first time that precise information flow tracking all the way down to the gates is a both possible and tractable.

References

[1] M. Tiwari, H. Wassel, B. Mazloom, S. Mysore, F. Chong, and T. Sherwood. Complete information flow tracking from the gates up. In *Proceedings of the 14th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, 2009.

Single-Shot, Geometry-Invariant Shadow Demultiplexing

Daniel A. Vaquero
Four Eyes Lab, UCSB
daniel@cs.ucsb.edu

Ramesh Raskar
MIT Media Lab
web.mit.media.edu/~raskar

Rogerio S. Feris
IBM Research
rsferis@us.ibm.com

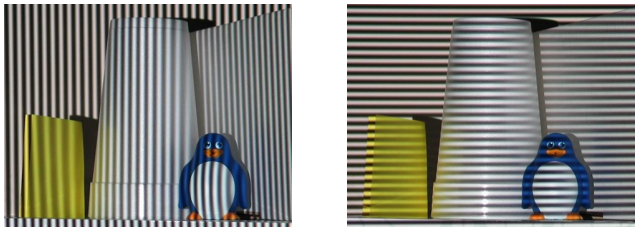
Matthew Turk
Four Eyes Lab, UCSB
mturk@cs.ucsb.edu

1. INTRODUCTION

In computer vision, many active illumination techniques employ projector-camera systems to facilitate the extraction of useful information from scenes. These approaches usually rely on the careful choice of an illumination pattern to be projected onto the objects. The image captured by the camera is a function of the projected pattern and its interaction with the objects in the scene; as the projected pattern is known, it is possible to exploit this information to recover properties of the imaged scene. Figure 1(a) illustrates this point. It shows objects imaged under the illumination of a projector placed at the left hand side of the camera. The projector sends equally spaced vertical white stripes onto the scene, but the observed patterns in the image vary according to the local orientation of the objects. Their local frequencies are useful cues for recovering surface orientation.

In this work, we follow an opposite direction. Rather than exploiting variations in the projected patterns due to depth and orientation changes, we show how a projector-camera setup can be built in a way that the imaged patterns are the same across the scene, no matter what the geometry of the objects is (see Figure 1(b)). Our method is based on a strategic alignment of the projector-camera configuration, and on a particular choice of the projected patterns. The technique is derived from a key observation from the geometrical relationship between two cameras, and the fact that a projector can be understood as a dual of a camera, sharing the same geometric characteristics [3].

We then demonstrate the usefulness of this property for demultiplexing shadows simultaneously generated by multiple projectors. The goal is to, given a single image of a scene illuminated by multiple projectors, segment the regions in shadow, and, for each shadowed region, determine which projectors caused it. In our method, each projector projects a distinct pattern onto the scene (each pattern has a different frequency). For a given projector, the regions in shadow will not contain the projected pattern, while the pattern will be present on the illuminated regions. By analyzing the frequencies of the patterns in the captured image, the shadow regions for each projector can be extracted. In this process, observing patterns with frequencies invariant to the scene geometry (Figure 1(b)) is desirable, as the frequencies to be searched for would be fixed and known; on the other hand, the scenario in Figure 1(a) greatly complicates the demultiplexing process in the presence of multiple projectors.



(a) (b)

Figure 1: The frequency of observed patterns is sensitive to the geometry of the scene. (a) Vertical stripes projected from a projector placed at the left hand side of the camera. Notice the different frequencies on the two slanted planes (left and right), and curved lines on the penguin; (b) By projecting horizontal stripes, the frequency of the observed patterns is geometry-invariant.

The method can be employed to extend the applicability of techniques that rely on the analysis of shadows cast by multiple light sources placed at different positions (such as [4]). Those methods usually take multiple pictures of the scene at different instants of time, with only one light source being triggered during the capture of each image. This often brings limitations for scenes with moving objects, as the images captured at different instants of time exhibit misaligned features. Our approach can be employed to obtain the shadows cast from multiple projectors with a single shot, enabling the processing of dynamic scenes.

In this extended abstract, we summarize the main ideas for building a projector-camera setup that achieves geometry-invariant frequency projection, and give an overview of a method for demultiplexing shadows from a single image. The reader is referred to [5] for more details and discussion.

2. GEOMETRY-INVARIANT FREQUENCY PROJECTION

Consider a projector-camera setup composed of a perspective camera and a perspective projector, which projects a sinusoidal pattern $f(x, y) = \frac{h}{2}[1 + \cos(\omega_1 x + \omega_2 y)]$ onto the scene, where ω_1 and ω_2 denote the angular frequencies of the sinusoid, in radians per pixel, and h is the amplitude of the mask.

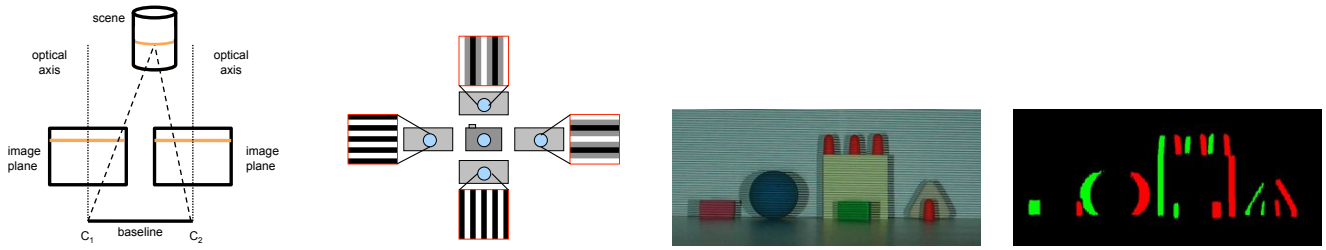


Figure 2: (a) correspondence between lines in the canonical stereo configuration, considering perspective cameras; (b) sample configuration with a camera and four projectors; (c) image taken using a camera and two projectors; (d) segmented shadows (red = left projector, green = right projector).

Now consider two perspective cameras pointing at the same direction, having parallel optical axes orthogonal to the baseline (the line that connects the centers of projection of both cameras), which is aligned with the horizontal coordinate axis. This configuration is known in the stereo vision literature as the *canonical configuration* [2]. In this arrangement, a row in one of the image planes corresponds to the same row in the other image plane. Figure 2(a) illustrates this. If we replace one of the cameras by a perspective projector, the same result holds, since a projector has the same geometry [3]. Thus, if we project horizontal patterns from a projector aligned with the camera along the horizontal coordinate axis, each projected horizontal line will be imaged at the same horizontal line in the camera. Therefore, the frequency of the observed sinusoidal patterns will be insensitive to variations in the shape of the objects.

The above reasoning suggests that a projector-camera setup for which the frequency of the observed patterns is geometry-invariant can be built by strategically choosing two elements: the **projector-camera alignment** and the **projected patterns**. First, the projector should be placed in the same plane as the camera, such that they point at the same direction and their optical axes are parallel. Second, patterns parallel to the direction of alignment between the camera and the projector should be projected. For example, for a projector placed to the left or to the right of the camera, it is best to project sinusoids with $\omega_1 = 0$ (horizontal stripes); for a projector placed above or below the camera, it is preferable to use vertical patterns ($\omega_2 = 0$).

3. CODED SHADOW DEMULTIPLEXING

Let us now look into the problem of shadow demultiplexing in a multi-projector, single-camera setup. Suppose that all projectors send a distinct sinusoidal pattern onto the scene at the same time, and the goal is to analyze the frequencies of the observed patterns in the image taken by the camera in order to determine the regions being illuminated by each individual projector. As discussed in the introduction, if we use the geometry-invariant setup from the previous section, then the complexity of the problem is greatly reduced, as the observed frequencies are fixed and known. A basic setup for this purpose would consist of a camera and multiple projectors placed in the same plane, such that each projector-camera pair satisfies the conditions described in the previous section. Also, the frequencies of the sinusoids should be distinct among the projectors that share the same direction of camera-projector alignment. Figure 2(b) illustrates an example of a setup that meets these requirements, by projecting sinusoids with frequencies of π and $\pi/2$ radians per pixel.

Given the single image captured using multiple projectors, the objective is to, for each projector p_i, determine the image regions that contain sinusoids with the frequency imprinted by p_i. A frequency-based texture segmentation algorithm can be applied to segment the regions based on the frequencies of the observed sinusoids. For simplicity, we used Gabor filters [1] for this purpose. These filters have high responses when a specific frequency is present in a given region of the image, and low responses otherwise. However, more sophisticated texture segmentation approaches could be used.

To illustrate, Figure 2(c) shows an image taken using a setup composed of a camera and two projectors. The projectors were placed to the left and right hand sides of the camera, projecting horizontal sinusoids with different frequencies. Notice the distinct patterns on the shadowed regions. Figure 2(d) shows the shadow demultiplexing results, where the red and green areas correspond to shadows determined to having had been cast by the left and right projectors, respectively.

4. REFERENCES

[1] A. Bovik, M. Clark, and W. Geisler. Multichannel texture analysis using localized spatial filters. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 12(1):55–73, Jan 1990.

[2] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition, 2004.

[3] R. Raskar and P. Beardsley. A self-correcting projector. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Kauai, Hawaii, 2001.

[4] R. Raskar, K. Tan, R. Feris, J. Yu, and M. Turk. A non-photorealistic camera: depth edge detection and stylized rendering using multi-flash imaging. *SIGGRAPH / ACM Trans. on Graphics*, 2004.

[5] D. A. Vaquero, R. Raskar, R. S. Feris, and M. Turk. A projector-camera setup for geometry-invariant frequency demultiplexing. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Miami, Florida, 2009.

Stranger: A String Analysis Tool for PHP Programs

Fang Yu
University of California, Santa Barbara
yuf@cs.ucsb.edu

Muath Alkhalaf
University of California, Santa Barbara
muath@cs.ucsb.edu

Tevfik Bultan
University of California, Santa Barbara
bultan@cs.ucsb.edu

ABSTRACT

Stranger is a string analysis tool for finding and eliminating security vulnerabilities in PHP applications. Given a program and an attack pattern (specified as a regular expression), Stranger automatically generates string-based vulnerability signatures, i.e., a characterization that includes all malicious inputs that can be used to generate attacks. Stranger uses an automata-based string analysis framework. Using forward reachability analysis it computes an over-approximation of all possible values that string variables can take at each program point. Intersecting these with the attack pattern yields the potential attack strings if the program is vulnerable. Using backward analysis, Stranger computes an over-approximation of all possible inputs that can generate those attack strings. In addition to identifying existing vulnerabilities and their causes, these vulnerability signatures can be used to filter out malicious inputs.

1. INTRODUCTION

Web applications provide critical services over the Internet and frequently handle sensitive data. Unfortunately, Web application development is error prone and results in applications that are vulnerable to attacks by malicious users. According to the Open Web Application Security Project (OWASP)'s top ten list that identifies the most serious web application vulnerabilities, the top three vulnerabilities are: 1) Cross Site Scripting (XSS), 2) Injection Flaws (such as SQL injection) and 3) Malicious File Execution. All these vulnerabilities are due to improper string manipulation. Programs that propagate and use malicious user inputs without sanitization or with improper sanitization are vulnerable to these well-known attacks.

We developed a string analysis tool that 1) identifies if a web application is vulnerable to attacks related to string manipulation, and 2) if it is vulnerable, generates a characterization of user inputs that might exploit that vulnerability. Such a characterization is called a vulnerability signature. Vulnerabilities related to string manipulation can be characterized as attack patterns, i.e., regular expressions that specify vulnerable values for sensitive operations (called sinks).

Given an application, vulnerability analysis identifies if there are any input values that could lead to a vulnerable value to be passed to a sensitive operation. Once a vulnerability is identified, the next important question is what set of input values can exploit the given vulnerability. A vulnerability signature is a characterization of all such input values. It can be used to identify how to sanitize the user input to eliminate the identified vulnerability, or to dynamically monitor the user input and reject the values that can lead to an exploit.

We use automata-based string analysis techniques [4] for vulnerability analysis and vulnerability signature generation. Our tool takes an attack pattern specified as a regular expression and a PHP program as input and 1) identifies if there is any vulnerability based on the given attack pattern, 2) generates a DFA characterizing the set of all user inputs that may exploit the vulnerability.

Our string analysis framework uses deterministic finite automaton (DFA) to represent values that string expressions can take. At each program point, each string variable is associated with a DFA. To determine if a program has any vulnerabilities, we use a forward reachability analysis that computes an over-approximation of all possible values that string variables can take at each program point. Intersecting the results of the forward analysis with the attack pattern gives us the potential attack strings if the program is vulnerable. The backward analysis computes an over-approximation of all possible inputs that can generate those attack strings. The result is a DFA for each user input that corresponds to the vulnerability signature.

We used Stranger to analyze four real-world web applications. Our results demonstrate that our tool can detect vulnerabilities in Web applications and identify the corresponding vulnerability signatures.

2. STRANGER TOOL

We implemented our approach in a tool called *Stranger* (STRing AutomatoN GEneratoR) that analyzes PHP programs. Stranger uses the front-end of Pixy, a vulnerability analysis tool for PHP that is based on taint analysis [2]. Stranger also uses the automata package of MONA tool [1] to store the automata constructed during string analysis symbolically. Stranger takes a PHP program as input and automatically analyzes it and outputs the possible XSS and SQL injection vulnerabilities in the program. For each input that leads to a vulnerability, it also outputs an automaton in a dot format that characterizes all possible string values for this input which may exploit the vulnerability, i.e., it outputs the vulnerability signatures.

The architecture of Stranger is shown in Figure 1. The tool consists of three parts: 1) The *PHP Parser* which parses the PHP code and constructs a control flow graph (CFG). 2) The *Taint Analyzer* which performs alias and dependency analyses, builds the dependency graphs, analyzes them and outputs tainted ones in which tainted user input is not prop-

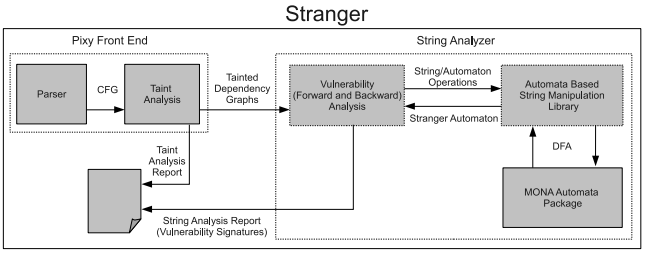


Figure 1: The Architecture of Stranger

erly sanitized. 3) The *String Analyzer* implements vulnerability (forward and backward) analysis on dependency graphs (as described in the previous section) for all sensitive sinks that are found to be tainted by taint analysis. If a sink is found to be secure by the string analyzer (with respect to the specified attack pattern), then it is guaranteed to be secure. If a sink is found to be vulnerable, then backward analysis computes the vulnerability signature. For a more detailed explanation you can refer to [3].

3. EXPERIMENTS

We experimented with Stranger on a number of benchmarks extracted from known vulnerable web applications: (1) `MyEasyMarket-4.1` (a shopping cart program), (2) `PBLguestbook-1.32` (a guestbook application), (3) `BloggIT-1.0` (a blog engine), and (4) `proManager-0.72` (a project management system). The taint analyzer automatically generates the tainted dependency graphs and identifies that all of them may be vulnerable.

In our experiments, we used an Intel machine with 3.0 GHz processor and 4 GB of memory running Ubuntu Linux 8.04. We use 8 bits to encode each character in ASCII. The performance of our vulnerability analysis is shown in Table 1. The backward analysis dominates the execution time from 77% to 96%. Taking a closer look, Table 2 shows the frequency and execution time of each of the string manipulating functions. preConcat (including prefix and suffix) consumes a large portion, particularly for (4) `proManager-0.72` that has a large size of constant literals involved. One reason is generating concatenation transducers during the computation. Note that the transducer has 3-tracks and uses 24 bits to encode its alphabet. On the other hand, our computation does not suffer exponential blow-up as expected for explicit DFA representation. This shows the advantage of using symbolic DFA representation (provided by the MONA DFA library), in which transition relations of the DFA are represented as Multi-terminal Binary Decision Diagrams (MBDDs).

	total time(s)	fwd time(s)	bwd time(s)	mem(kb)
1	0.569	0.093	0.474	2700
2	3.449	0.124	3.317	5728
3	1.087	0.248	0.836	18890
4	16.931	0.462	16.374	116097

Table 1: Total Performance

Finally, Table 3 shows the data about the DFAs that Stranger generated. Reachable Attack is the DFA that accepts all possible attack strings at the sink node. Vulnerability Signature is the DFA that accepts all possible malicious inputs that can exploit the vulnerability. We closely look at

	CONCAT	REPLACE	preCONCAT	preREPLACE
	#operations/time(s)			
1	6/0.015	1/0.004	2/0.411	1/0.004
2	19/0.082	1/0.004	11/3.166	1/0.0
3	22/0.038	4/0.112	2/0.081	4/0.54
4	14/0.014	12/0.058	26/11.892	24/3.458

Table 2: String Function Performance

the vulnerability signature of (1) `MyEasyMarket-4.1`. The signature actually accepts α^* `<`α^* `s`α^* `c`α^* `r`α^* `i`α^* `p`α^* `t`α^* with respect to the attack pattern Σ^* `<script`Σ^*. α is the set of characters, e.g., `!`, that are deleted in the program. An input such as `<!script` can bypass the filter that rejects Σ^* `<script`Σ^* and exploit the vulnerability. This shows that simply filtering out the attack pattern can not prevent its exploits. On the other hand, the exploit can be prevented using our vulnerability signature instead.

It is also worth to note that both vulnerability signatures of (2) `PBLguestbook-1.32` accept arbitrary strings. By manually tracing the program, we find that both inputs are concatenated to an SQL query string which is used in one sink without proper sanitization. Since an input can be any string, the pre-image of one input is the prefix of Σ^* `OR '1'='1'`Σ^* that is equal to Σ^*, while the pre-image of another input is the suffix of Σ^* `OR '1'='1'`Σ^* that is also equal to Σ^*. This case shows a limitation in our approach. Since we do not model the relations among inputs, we can not specify the condition that one of the inputs must contain `OR '1'='1'`.

	Reachable Attack (Sink)		Vulnerability Signature (Input)	
	#states	#bdd nodes	#states	#bdd nodes
1	24	225	10	222
2	66	593	2	9
2	66	593	2	9
3	29	267	92	983
4	131	1221	57	634
4	136	1234	174	1854
4	147	1333	174	1854

Table 3: Attack and Vulnerability Signatures

4. REFERENCES

[1] BRICS. The MONA project. http://www.brics.dk/mona/.

[2] Nenad Jovanovic, Christopher Krügel, and Engin Kirda. Pixy: A static analysis tool for detecting web application vulnerabilities (short paper). In *Proceedings of the 2006 IEEE Symposium on Security and Privacy (S&P 2006)*, pages 258–263, 2006.

[3] F. Yu, M. Alkhalaf, and T. Bultan. Generating vulnerability signatures for string manipulating programs using automata-based forward and backward symbolic analyses. Technical Report 2009-11, UCSB, 2009.

[4] Fang Yu, Tevfik Bultan, Marco Cova, and Oscar H. Ibarra. Symbolic string verification: An automata-based approach. In *Proceedings of the 15th International SPIN Workshop on Model Checking Software (SPIN 2008)*, pages 306–324, 2008.

TRUST: A General Framework for Truthful Double Spectrum Auctions

Xia Zhou and Heather Zheng
Department of Computer Science
University of California, Santa Barbara
{xiazhou, htzheng}@cs.ucsb.edu

1. INTRODUCTION

Due to their perceived fairness and allocation efficiency, auctions are among the best-known market-based mechanisms to distribute spectrum. In a well-designed auction, everyone has an equal opportunity to win and the spectrum is sold to bidders who value it the most. In the past decade, the FCC (Federal Communications Commission) and its counterparts across the world have been using *single-sided* auctions to assign spectrum to wireless service providers in terms of *predetermined* national/regional long-term leases.

In this work, we show that auctions can be designed to dynamically redistribute spectrum across multiple parties to meet their own demands. That is, the auctioneer runs *double* spectrum auctions to enable multiple sellers and buyers to trade spectrum dynamically. In this way, existing spectrum owners (as sellers) can obtain financial gains by leasing their selected idle spectrum to new spectrum users; new users (as buyers) can access the spectrum they desperately need and in the format they truly desire. By multiplexing spectrum supply and demand in time and space, dynamic auctions can significantly improve spectrum utilization.

In addition to enabling dynamic trading, our proposed spectrum auctions recognize that spectrum is reusable among bidders. Exploiting such reusability improves auction efficiency, yet the reusability also makes spectrum fundamentally different from conventional goods (*e.g.* paintings and bonds), and introduces significant difficulties in designing economic-robust auctions as shown by examples in [6, 8].

To address these challenges, we propose a framework for TRuthful doUble Spectrum aucTions (TRUST), achieving truthfulness and enabling spectrum reuse in double auctions. TRUST takes as input any reusability-driven spectrum allocation algorithm, and applies a novel winner determination and pricing mechanism to select winning sellers and buyers. To our best knowledge, TRUST is the first framework for truthful double spectrum auctions with spectrum reuse.

2. PROBLEM MODEL & CHALLENGES

We consider a single-round, sealed-bid, and collusion-free double spectrum auction with one auctioneer, M sellers, and N buyers. Each seller contributes one distinct channel and each buyer only requests one channel. To model a double spectrum auction, we define the following notations for both sellers and buyers: For a seller m, B_m^s is its bid, the minimum payment required to sell a channel; V_m^s is its true valuation of the channel; P_m^s is the actual payment received if it wins the auction; and its utility is $U_m^s = P_m^s - V_m^s$ if it wins the auction, and 0 otherwise; For a buyer n, B_n^b is its bid, the maximum price it is willing to pay for a channel; V_n^b is its true valuation of a channel; P_n^b is the price it pays if it wins the auction, and its utility is $U_n^b = V_n^b - P_n^b$ if it wins, and 0 otherwise. Then we can further define the three important economic properties as follows:

1). Truthfulness

DEFINITION 1. *A truthful auction is one in which no buyer n /seller m can obtain higher utility U_n^b/U_m^s by setting $B_n^b \neq V_n^b/B_m^s \neq V_m^s$.*

Truthfulness is essential to resist market manipulations and ensure auction fairness and efficiency.

2). Individual Rationality

DEFINITION 2. *A double auction is* individual rational *if no winning buyer pays more than its bid (*i.e. $P_n^b \leq B_n^b$*), and no winning seller gets paid less than its bid (*i.e. $P_m^s \geq B_m^s$*).*

This property ensures non-negative utilities for bidders who bid truthfully, providing them the incentives to participate.

3). Budget Balance

DEFINITION 3. *A double auction is* ex-post budget balanced *if the auctioneer's profit $\Phi \geq 0$. The profit is the difference between the revenue collected from buyers and the expense paid to sellers: $\Phi = \sum_{n=1}^{N} P_n^b - \sum_{m=1}^{M} P_m^s \geq 0$.*

This property ensures that the auctioneer has incentives to set up the auction.

Existing double auction designs	**Spectrum Reuse**	**Truthfulness**	**Ex-post Budget Balance**	**Individual Rationality**
VCG	X	√	X	√
McAfee	X	√	√	√
VERITAS extension	√	X	√	√
TRUST	√	√	√	√

To enable efficient spectrum trading, the auction design must exploit spectrum reusability to improve spectrum utilization and achieve the three critical economic properties. The reusability, however, brings significant difficulties in achieving economic robustness. As highlighted by the above table, conventional truthful double auction designs (VCG [1] and McAfee [2]) do not consider reusability. Prior work on truthful spectrum auctions (VERITAS [6]) only addresses buyer-only auctions, and [8] further shows that the extension of

single-sided auction to double auctions fails to be truthful. We refer readers to [8] for the detailed examples.

3. TRUST: DESIGN

TRUST [7] breaks the barrier between spectrum reuse and economic-robustness in double spectrum auctions. In essence, it enables spectrum reuse by applying a spectrum allocation algorithm to form buyer groups. It achieves the three economic properties via the bid-independent group formation and a reusability-aware pricing mechanism. TRUST consists of three components:

Grouping Buyers. TRUST groups multiple non-conflicting buyers into groups so that buyers in each group do not conflict and can reuse the same channel. This is done privately by the auctioneer performing a spectrum allocation algorithm and grouping buyers assigned to the same channel to a group. Unlike VERITAS, the group formation is independent of the buyer bids to prevent bidders from rigging their bids to manipulate its group size and members.

The group formation can cope with various interference models by using different spectrum allocation algorithms. If the buyer interference condition is modeled by a conflict graph, the group formation is equivalent to finding the independent sets of the conflict graph [3, 5]. If the condition is modeled by the physical Signal to Interference and Noise Ratio (SINR) [4], TRUST finds multiple sets of buyers who can transmit simultaneously and maintain an adequate received SINR. Assuming channels are homogeneous, TRUST performs this allocation only to form buyer groups, not to assign specific channels to buyers.

Determining Winners. Next, TRUST treats each buyer group as a super-buyer and runs the conventional double spectrum auction algorithm to determine the winning sellers and super-buyers. Let $G_1, G_2, ..., G_L$ represent the L groups formed. For any group G_l with $n_l = |G_l|$ buyers, the group bid π_l is:

$$\pi_l = \min\{B_n^b | n \in G_l\} \cdot n_l. \quad (1)$$

TRUST sorts the seller bids in non-decreasing order and the buyer group bids in non-increasing order: $\mathbb{B}' : B_1^s \leq B_2^s \leq ... \leq B_M^s$, and $\mathbb{B}'' : \pi_1 \geq \pi_2 \geq ... \geq \pi_L$. Define k as the last profitable trade:

$$k = \underset{l \leq \min\{L,M\}}{\text{argmax}} \ \pi_l \geq B_l^s. \quad (2)$$

Then the auction winners are the first $(k-1)$ sellers, and the first $(k-1)$ buyer groups.

Pricing. To ensure truthfulness, TRUST pays each winning seller m by the kth seller's bid B_k^s, and charges each winning buyer group l by the kth buyer group's bid π_k. This group price is evenly shared among the buyers in the group:

$$P_n^b = \pi_k / n_l, \ \ \forall n \in G_l. \quad (3)$$

No charges or payments are made to losing buyers and sellers. The uniform pricing is fair because buyers in a winning group obtain the same channel, thus should be charged equally. In addition, to ensure individual rationality, a group bid must not exceed the product of the lowest buyer bid in the group and the number of buyers in the group, which is used in the process of determining winning groups. With such pricing mechanism, the auctioneer's profit becomes $\Phi = (k-1) \cdot (\pi_k - B_k^s)$ and it is easy to show that $\Phi \geq 0$.

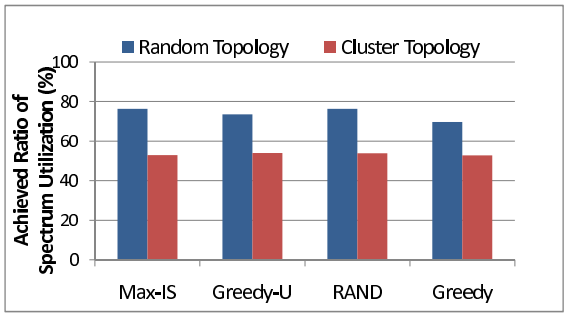


Figure 1: The percentage of spectrum utilization achieved by TRUST comparing to that of pure allocations without economic factors. Four various allocation algorithms are considered including Max-IS, Greedy-U, Greedy, and random allocation RAND.

4. TRUST: PERFORMANCE

On the other hand, ensuring these economic properties comes at a cost in spectrum utilization. This is because TRUST selects winning buyer groups by the minimum bid in the group multiplied by the group size, so that groups of different sizes have equal opportunity in being chosen. However, the conventional spectrum allocation algorithms always choose large groups, leading to an advantage in spectrum utilization. Figure 1 illustrates the ratio of TRUST's spectrum utilization over that of conventional spectrum allocations without economic consideration [3, 5]. It includes TRUST with four spectrum allocation algorithms, and examines the performance using random and clustered topologies. In random network topologies, TRUST achieves 70–80% spectrum utilization of the conventional spectrum allocation, while in clustered topologies, TRUST sacrifices roughly 50% of spectrum utilization for economic robustness. This is because in clustered topologies, the group sizes become much more diverse, and hence TRUST is more likely to select a set of small buyer groups which degrades the overall spectrum utilization. Note that finding the optimal buyer group formation is an interesting topic to further explore, and it would depend on conflict constraints and bid distribution.

5. REFERENCES

[1] Babaioff, M., and Nisan, N. Concurrent auctions across the supply chain. In *Proc. of Economic Commerce* (2001).

[2] McAfee, R. P. A dominant strategy double auction. *Journal of Economic Theory 56*, 2 (April 1992), 434–450.

[3] Ramanathan, S. A unified framework and algorithm for channel assignment in wireless networks. *Wirel. Netw. 5*, 2 (1999), 81–94.

[4] Reis, C., Mahajan, R., Rodrig, M., Wetherall, D., and Zahorjan, J. Measurement-based models of delivery and interference in static wireless networks. In *Proc. of SIGCOMM* (September 2006).

[5] Subramanian, A. P., Gupta, H., Das, S. R., and Buddhikot, M. M. Fast spectrum allocation in coordinated dynamic spectrum access based cellular networks. In *Proc. of IEEE DySPAN* (November 2007).

[6] Zhou, X., Gandhi, S., Suri, S., and Zheng, H. eBay in the sky: Strategy-proof wireless spectrum auctions. In *Proc. of MobiCom* (Sept. 2008).

[7] Zhou, X., and Zheng, H. TRUST: A general framework for truthful double spectrum auctions. In *Proc. of INFOCOM* (April 2009).

[8] Zhou, X., and Zheng, H. TRUST: A general framework for truthful double spectrum auctions (extended). *UCSB Technical Report* (2009).

http://gswc.cs.ucsb.edu/