CS 668 - Spring ‘06

Student Research Presentations
(RB 122)

Date/Time	Presenter	Title
Monday, April 10 4:00 pm	Fabien Poulard	Utility of random generated graphs to optimize peer-to-peer networks
Monday, April 10 4:30 pm	Todd Chaffins	Performance Comparison of Vertex Coloring Algorithms on CPUs and GPUs
Wednesday, April 12 4:00 pm	James Haberly	Graph-theoretic image processing techniques
Wednesday, April 12 4:30 pm	Mahbub Majumder	Graceful labeling of trees
Monday, April 17 4:00 pm	Hsiaoying Su	Path-finding - An application of graph theory in computer games
Monday, April 17 4:30 pm	Mandeep Singh Atwal	Tree Vertex Splitting Problem - Applications to Distribution Networks
Wednesday, April 19 4:00 pm	Bryan Ritz	Analysis of Techniques Used in Drawing Graphs with Subgraphs (with Case Study)

See the following pages for abstracts.

Monday, April 10
4:00 pm

Utility of random generated graphs to optimize peer-to-peer networks

Fabien Poulard

The first peer-to-peer networks emerged with the Gnutella protocol in the early 2000, thanks to a team of developers at Nullsoft. When tested in a lab, the protocol was able to manage some hundreds nodes; but when released, it was quickly used by thousands of users, creating as much nodes and revealing the weaknesses of its conception. Since then the peer-to-peer has evolved and some others protocols have appeared, fixing partially the weaknesses of Gnutella. However, those protocols do not really implement a real peer-to-peer network, but a kind of topologically hybrid network sharing characteristics of the decentralized topology and some characteristics from other topologies (most likely hierarchy and centralized).

As at the origins the peer-to-peer was developed to share music files (Nullsoft develops a widely used mp3 player product) “illegally”, there was no more research to extend the capabilities and exploit the peer-to-peer networks. However, nowadays, peer-to-peer is considered as a possible topology for a lot of networks, and especially for the mobile device applications. So, improving the algorithms at the origin of Gnutella becomes industrially interesting.

Much progress has been achieved in the last few years, especially due to graph theory. The main issues to resolve are about preserving the connectivity of the graph when adding or removing a peer and in making the graph evolve towards an expander graph to optimize the exchanges between clients. However, the huge constraint is to make that happen with a minimum number of transactions considering that each manipulation of the network implies latencies and loss of packets that will have to be resent. Another issue is the fault tolerance, as each node can fail due to the extreme heterogeneity of the network, it must always be possible to find another path from one node to another.

The different techniques I will try to cover during the presentation are based on the random transformation of regular graphs and the de Bruijn graphs. We will see how they can be applied to the original Gnutella to improve its efficiency without losing its decentralized characteristics.

References

• *Peer-to-peer Networks based on Random Transformations of Connected Regular Undirected Graphs*, P. Mahlmann, C. Schindelhauer, ACM July 2005
• *Graph-Theoretic Analysis of Structured Peer-to-Peer Systems : Routing Distances and Fault Resilience*, D. Loguinov, J. Casas, X. Wang, IEEE/ACM Transactions on Networking • *Distributed Construction of Random Expander Networks*, C. Law, K. Siu, IEEE INFOCOM
• *The diameter of random massive graphs*, L. Lu, Proceedings of the twelfth annual ACMSIAM symposium on Discrete algorithms
• *Generating Random Regular Graphs*, J.H. Kim, V.H. Vu, Proceedings of the thirty-fifth annual ACM symposium on Theory of computing
• *On the Fundamental Tradeoffs Between Routing Table Size and Network Diameter in Peer-to-peer Networks*, J. Xu, IEEE INFOCOM

Monday, April 10
4:30 pm

Performance Comparison of Vertex Coloring Algorithms on CPUs and GPUs

Todd Chaffins

Current high-end graphics processing units (GPUs) have evolved from dump co-processors with fixed functionality into highly capable stream processors. [Purcell, et al. 2002] In the ever-growing games industry there is a demand to be able to create more realistic graphics and effects. The current trend is to create these realistic graphics and effects through the use of graphics code known as shaders [Harris, et al. 2003]. Shaders are small pieces of code which determine how graphics are rendered on the screen. These shaders and their adoption in games require the graphics card manufacturers to create faster and more programmable GPUs. This pressure will continue and as such the speed and programmability of GPUs are set to continue to increase as time goes on. [Christen 2005]

This increase in processing power and the parallel nature of GPUs has led to the adoption of the GPU for general purpose computations outside of the realm of graphics. While the GPU has made vaster architectural changes over recent years CPUs have also made advances. Aside from the common increases associated with CPUs (clock speed and cache), CPUs have moved to be more parallel with a dual-core architecture. With parallel vertex-coloring algorithms [Kale, et al. 1995] the question is raised as to which approach will yield the fastest results when performing vertex-coloring: CPU based, GPU based, or a hybrid CPU/GPU approach. This research seeks to implement, instrument, and measure the performance of these approaches in a quantitative manner and evaluate the economy of these approaches.

References:

[Purcell, et al. 2002] Purcell, T.J., Buck, I., Mark, W.R. and Hanrahan, P. Ray Tracing on Programmable Graphics Hardware. In *Proceedings of SIGGRAPH 2002*, ACM / ACM Press. 2002.

[Christen 2005] Christen M.: Ray Tracing on GPU. Diploma thesis, University of Applied Sciences Basel, Switzerland, 2005.

[Kale, et al. 1995] L. V. Kale, B. H. Richards, and T. D. Allen. Efficient parallel graph coloring with prioritization. In *Lecture Notes in Computer Science*, volume 1068, pages 190{208. Springer-Verlag, August 1995.

[Harris, et al. 2003] Mark Harris, Greg James, Physically-Based Simulation on Graphics Hardware, *GameDevelopers Conference*, 2003.

Wednesday, April 12
4:00 pm

Graph-theoretic image processing techniques
James Haberly

The aim of my proposed topic would be to present an exposition on Graph Theoretic approaches and algorithms as they're being researched for use in image processing and machine vision. Some example problems in the area of image processing and machine vision are; computational complexity, object recognition, object measurement, image segmentation, edge detection, noise detection and filtering, line, arc and other feature detection, image coding and compression.

The project will be focused on presenting how graph theoretic algorithms with examples such as the Prim and Kruskal algorithms for minimum spanning trees, Dijkstra's and Dial's shortest path and graph theoretic Euclidean distance mapping techniques are being examined for problem solving in the image processing and machine vision fields.

I do not want to yet limit the scope of this proposal by proposing an exposition on any one paper or problem since I'm just beginning the research phase of the project. A couple specific areas that may become the main focus of the project are:

1. Improvements in computational speed using graph-theoretic image processing techniques [1].

2. Object recognition using a graph theoretical approach [2].

The image processing and machine vision industry is a fast growing and exciting field. I'm looking forward to researching the topic further.

References:

[1] "Faster Graph-Theoretic Image Processing via Small-World and Quadtree Topologies" Leo Grady and Eric L. Schwartz Dept. of Imaging & Visualization, Siemens Corp. Res. Inc., Princeton, NJ, USA; This paper appears in: Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on Publication Date: 27 June-2 July 2004
Volume: 2, On page(s): II-360- II-365 Vol.2

[2] "Color Invariant Object Recognition using Entropic Graphs" Jan C. van Gemert, Gertjan J. Burghouts, Frank J. Seinstra, Jan-Mark Geusebroek Intelligent Systems Lab Amsterdam, Informatics Institute, University of Amsterdam, Kruislaan 403, 1098 SJ Amsterdam, The Netherlands.

[3] "Graph-Theoretical Methods in Computer Vision" Ali Shokoufandeh1 and Sven Dickinson2 G.B. Khosrovshahi et al. (Eds.): Theoretical Aspects of Computer Science, LNCS 2292, pp. 148–174, 2002.

Wednesday, April 12
4:30 pm

Graceful labeling of Trees
Mahbub Majumder

A tree T with *n* vertices is said to be gracefully labeled if its vertices are labeled with the integers [1..*n*] such that the edges, when labeled with the difference between their endpoint vertex labels, are uniquely labeled with the integers [1..*n*-1]. If T can be gracefully labeled, it is called a "graceful tree".

The concept of graceful labeling of trees and graphs was introduced by Rosa (1967). The term "graceful labeling" was invented by Golomb (Golomb 1972). The Graceful Tree Conjecture states that all trees are graceful. There have been over 670 papers to date on various graph labeling methods and issues (Gallian 2005). So far, no proof of the truth or falsity of the conjecture has been found. Even though the conjecture is open, some partial results have been proved (Gallian 2005).

My motivation in pursuing this project came from its nature and the study many people have put into it. My aim with this project work is to

1. Find out current works and results.
2. Study and understand the condition of gracefulness
3. If possible, add some ideas in graceful Labeling

References:

Gallian J. A., *A Dynamic Survey of Graph Labeling (2005)*, Electronic Journal of Combinatorics.

Golomb S.W. *How to number a graph*. In R.C. Read, editor, Graph Theory and Computing, pages 23-37. Academic Press, 1972.

Rosa A., *On certain valuations of the vertices of a graph*, Theory of Graphs (Internat. Symposium, Rome, July 1966), Gordon and Breach, N.Y. and Dunod Paris (1967) 349-355.

Monday, April 17
4:00 pm

Path-finding - An application of graph theory in computer games

Hsiaoying Su

Graph theory is widely used in solving and presenting computer games. One of the most common applications is path-finding. Path-finding algorithms grant agents in the virtual world the ability to consciously find their own way around the land. They can also be used in real life to find driving directions, such as the service offered by many popular web sites. The project is going to focus on the game implementation.

Path-finding algorithms are usually, but not only, used in computer games catalogued as role playing games (RPGs). Programmers build a virtual world for the games. The characters, or called autonomous agents, have certain level of artificial intelligence. The simplest way to present their intelligence is to find their own paths moving around the world without hitting a tree or going through a wall. The successful implementation of path-finding is important to the artificial intelligence performance of a game.

According to the numbers of sources and destinations, path-finding algorithms can be roughly divided into three categories: single source, single pair, and all pairs. In single source algorithms, the path from one node to all the others is required. However, single pair algorithms take a specific source and destination. Only one path is required in response. The all pairs algorithm returns the shortest paths from every node to all other nodes.

This project plans to explore different path-finding algorithms and their complexity. Furthermore, coding these algorithms in JEdit and practically experiment their efficiency. In single source algorithms, Dijkstra's Algorithm and Bellman-Ford-Moore, which deals with negative arc-lengths graph, would be discussed. Then, the most popular algorithm A* (A star) would be presented to implement single-pair shortest path finding. To find out all-pairs shortest paths, the algorithms described above could be used in a naïve but inefficient way. The project then would seek out whether there exist more efficient algorithms to solve the problem.

Path-finding related topics have been discussed on more application than research. However, there are still some interesting studies going on. One of a recent research is about solving incoherent behavior of multiple units in a cluttered environment. Another one is to discuss an open problem of emulating the rich complexity of real pedestrians in urban environment. The two papers are listed below as references.

References

- Kamphuis A., Overmars M. H.: Motion planning: Finding Paths for Coherent Groups using Clearance. In Eurographics/ACM SIGGRAPH Symposium on Computer Animation (2004).
- Shao W., Terzopoulos D.: Artificial intelligence for animation: Autonomous pedestrians. In Eurographics/ACM SIGGRAPH Symposium on Computer Animation (2005).

Monday, April 17
4:30 pm

Tree Vertex Splitting Problem - Applications to Distribution Networks

Mandeep Singh Atwal

In an Ethernet network, the number of connections (taps) and their intervening distances are limiting factors [BN90]. Repeaters are used to regenerate the signal every 500 meters or so [BN90]. If these repeaters were not used, “standing waves” (additive reflections) would distort the signal and cause errors [BN90]. Because collision detection depends partly on timing, only five 500-meter segments and four repeaters can be placed in series before the propagation delay becomes longer than the maximum allowed time period for detecting a collision [BN90].

Directed acyclic graphs (dags) or directed trees can be used to model such interconnection networks. Each edge of such a tree is labeled with a real number called its weight. Trees with edge weights are called *weighted trees*. Nodes or vertices in the tree correspond to receiving stations and edges correspond to transmission lines [HSR98]. Each edge weight corresponds to the delay in traversing that edge [HSR98]. However, as stated above, the network may not be able to tolerate losses in signal strength beyond a certain level.

In places where the loss exceeds the tolerance level, repeaters have to be placed. Given a network and a loss tolerance level *Tree Vertex Splitting Problem* is to determine an optimal placement of repeaters.

RESEARCH OBJECTIVES

The proposed research aims at the study of *Tree Vertex Splitting Problem (TVSP).* Designing the algorithms for *TVSP* and analyzing in terms of the computing time and space requirements. The most efficient algorithm can then possibly be implemented in C++/Java.

However, it is not an objective to implement the algorithm for the proposed study.

REFERENCES

[BN90] Barry Nance, *Network Programming in C*, QUE Corporation 1990, ISBN: 0-88022-569-6, page # 23.

[HSR98] Ellis Horowitz, Sartaj Sahni, Sanguthevar Rajasekaran, *Fundamentals of Computer Algorithms*, Galgotia Publications’ Pvt. Ltd. 1998, ISBN: 81-7515-257-5, page # 203.

[PRS98] Doowon Paik, Sudhakar Reddy, Sartaj Sahni, *Vertex Splitting In Dags And Application To Partial Scan Designs And Lossy Circuits,* International Journal of Foundations of Computer Science, 1998.

[ME93] Matthias Mayer, Fikret Ercal, *Genetic Algorithms for Vertex Splitting in DAGs,* Proceedings of the 5th International Conference on Genetic Algorithms, 1993, ISBN: 1-55860-299-2

[SR96] Stephanie Forrest, *Genetic Algorithms,* ACM Computing Surveys, Vol. 28, No. 1, March 1996.

Wednesday, April 19
4:00 pm

Analysis of Techniques Used in Drawing Graphs with Subgraphs
(with Case Study)
Bryan Ritz

In the paper "Drawing Graphs Within Graphs" [5] by Paul Holleis, Thomas Zimmermann, and Daniel Gmach, the authors present methods for helping to reduce complexity of large and complicated graphs and subgraphs. The methods of finding an optimal layout of subgraphs and a summary graph, of the use of connection sets, and of the use of motifs are all combined into an approach for emphasizing subgraphs within graphs. This paper will attempt to evaluate the worthiness of these methods through examination of sources used in the paper and by applying the methods to a case study in the form of a complex graph (displayed using JEdit).

References:

1. F. J. Brandenburg. Graph clustering 1: Cycles of cliques. In *Proceedings of the Graph Drawing 1997*, volume 1353 of *Lecture Notes in Computer Science,* Berlin, Germany, 1997. Springer.

2. G. Di Battista, P. Eades, R. Tamassia, and I. G. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice-Hall, Englewood Cliffs, N.J., 1999.

3. T. M. J. Fruchterman and E. M. Reingold. Graph drawing by force-directed placement. *Software-Practice and Experience*, 21(11):1129-1164, 1991.

4. T. Kamada and S. Kawai. An algorithm for drawing general undirected graphs. *Information Processing Letters*, 31(1):7-15, 1989.

5. P. Holleis, T. Zimmermann, D. Gmach. Drawing Graphs Within Graphs. *Journal of Graph Algorithms and Applications*, 9(1):7-18, 2005.

Case Study will use TouchGraph as applied to cs.bsu.edu.
1) Go to http://www.touchgraph.com/TGGoogleBrowser.html
2) Enter "cs.bsu.edu" without the quotes in the text field and hit enter.