

Statement of Research Interests Kamesh Madduri (kamesh@cc.gatech.edu)

Emerging large-scale Informatics applications motivate and shape my research agenda. I aspire to solve challenging problems arising in social and technological network analysis (e.g., identification of implicit online communities, viral marketing strategies, quantifying centrality and influence in interaction networks, web algorithmics), systems biology (for instance, interactome analysis, epidemiological studies, disease modeling), and homeland security (e.g., detecting trends, anomalous patterns from socio-economic interactions and communication data). Specifically, my research goals are to design efficient algorithms and develop high-performance software for enabling novel, insightful network analyses.

Data sources and applications from online social networking, the financial sector, surveillance, and scientific computing generate massive amounts of information; further, the data varies in the quality and type of content, and is dynamically generated. Data streaming algorithms have emerged as a popular approach for statistical analysis of large data sets. Complementarily, a graph or network representation is a convenient abstraction in many cases – unique data entities are represented as vertices, and the interactions between them are depicted as edges. The vertices and edges can further be typed, classified, or assigned attributes based on relational information from the heterogeneous sources. Common analysis queries on the data set can now be naturally expressed as variants of problems related to graph connectivity, flow, or partitioning. For tractable analysis of massive data sets, we need holistic approaches that supplement graph algorithms with techniques from statistics, optimization, machine learning and classical web and social science algorithms. Parallel computing is a key enabling technology for massive-scale informatics, and my specific research contributions are on the *design of novel parallel graph algorithms*, *data-centric algorithm engineering*, and *efficient software implementations* for problems dealing with centrality, connectivity, and community identification in large-scale networks.

1. Massive-scale Social and Information Network Analysis

Real-world systems such as the Internet, socio-economic interactions, and biological networks have been extensively studied from an empirical perspective, and this has led to the development of a variety of models to understand their topological properties and evolution. In particular, some of the common structural features they exhibit include a low graph diameter, an unbalanced degree distribution, self-similarity, and the presence of dense sub-graphs. Analogous to the small-world (the presence of short paths, due to the low graph diameter) phenomenon, these real-world data sets are broadly referred to and modeled as *small-world* networks.

The analysis of massive-scale informatics networks poses several research challenges. First, since the topology of small-world networks is significantly different from previously-studied real-world graph abstractions (e.g., road networks, finite-element meshes from scientific computing), the algorithmic successes from these domains cannot be easily translated to small-world graph analysis. Second, the heterogeneous and dynamic nature of the data raises abstraction and representation questions, and classical graph algorithms cannot be directly applied to process this data. Finally, it is infeasible to do exact in-core computations on massive graphs (by *massive*, I refer to graphs where the number of vertices and edges are 100 million, or more) due to the limited physical memory on current workstations. While advanced supercomputing resources mitigate the memory requirements imposed by massive graph analysis, scalable parallel graph algorithm design is still a tough problem. Due to their low diameter and unbalanced degree distribution, small-world networks are often difficult to partition well. Parallel graph algorithms are typically memory intensive, and

the memory accesses are fine-grained and highly irregular. For achieving high performance, it is critical to design parallel approaches that fully exploit the underlying architectural features. Next I briefly summarize some of my research on novel parallel algorithms for graph traversal, centrality, and community identification problems. These approaches are optimized for shared memory architectures, and also exploit typical topological characteristics of small-world networks, such as the low graph diameter, sparse connectivity, and skewed degree distribution.

Graph Traversal and Connectivity. Breadth-first search (BFS) is a fundamental paradigm in the design of complex graph algorithms. BFS, and in general graph traversal, can be parallelized in one of the following two ways: *level-synchronously*, where vertices at each level are visited in parallel; or using *path-limited searches*, where multiple searches are concurrently executed and aggregated. The level-synchronous approach is particularly suitable for small-world networks due to their low graph diameter. It is also critical to minimize synchronization overhead in the design of parallel algorithms for graph-theoretic problems. We aggressively reduce locking and barrier constructs through algorithmic changes, as well as incorporate implementation optimizations. For BFS, we designed a lock-free, level-synchronous algorithm that significantly reduces shared memory contention. We implemented BFS and report performance on the Cray MTA-2, a massively multi-threaded parallel system with hardware support for fine-grained synchronization. For a scale-free graph of one billion edges, our BFS implementation on the MTA-2 is more than *two orders of magnitude* faster than the state-of-the-art external memory implementation [5]. In related work, for the Ninth DIMACS Implementation Challenge on Shortest Paths, we solved the single-source shortest paths problem using a level-synchronous approach on a small-world graph of 400 million vertices and two billion edges, the largest graph instance in the competition [10]. The parallel implementation takes about 10 seconds on a 40-processor MTA-2 system, with an absolute speedup of close to 30. The results demonstrate that the massively multithreaded paradigm, coupled with support for fine-grained synchronization, leads to exceptional performance for massive small-world network problems.

Centrality. We designed the *first parallel algorithms* for evaluating several centrality indices frequently used in complex network analysis [6]. Our implementations achieve impressive parallel performance on real-world datasets such as the web graph, protein-interaction networks, social relationship networks, and citation graphs. Consider, for instance, the evaluation of betweenness, a global centrality metric based on shortest-path enumeration. It is compute-intensive, with the work complexity quadratic in the number of vertices. We compute the exact betweenness centrality value of all the vertices in a large US patent citation network (3 million patents, 16 million citations) in 42 minutes, on 16 processors of an IBM p5 570 system. Our approaches enable rigorous centrality analysis of networks *three orders of magnitude* larger than instances that can be processed by commercial social network analysis (SNA) software packages.

Community Identification. A key problem in social network analysis is that of finding communities, dense components, or detecting other latent structure. Classical approaches based on the Kernighan-Lin algorithm, flow-based algorithms, and hierarchical clustering work well for specific classes of networks (e.g., abstractions from scientific computing, physical topologies), but perform poorly for small-world networks. We designed three new community identification schemes (two hierarchical agglomerative approaches, and one divisive clustering algorithm) [9] that optimize modularity, a SNA clustering measure. We also conducted an extensive experimental study and demonstrated that our parallel schemes give significant running time improvements over existing

modularity-based clustering heuristics. For instance, our novel divisive clustering approach based on approximate edge betweenness centrality [3] is *more than two orders of magnitude* faster than the popular Girvan-Newman community detection algorithm on the Sun Fire T2000 multicore system, while maintaining comparable clustering quality.

SNAP. Identifying common kernels and abstractions from our research on small-world graph algorithms, we have designed an open-source, parallel infrastructure for *massive small-world graph analysis* called SNAP (Small-world Network Analysis and Partitioning). SNAP is a modular graph infrastructure implemented in C, uses POSIX threads and OpenMP primitives for parallelization, and targets multicore and manycore platforms. Our intent with SNAP is to provide a simple and intuitive interface for network analysis and application design, hiding the parallel programming complexity from the user. In addition to path-based, centrality, and community identification queries on large-scale graphs, we support commonly-used preprocessing kernels and quantitative measures that consider the global network topology.

2. Scalable Multicore Algorithm Design

A broader research theme encompassing my work is the design of scalable algorithms and software for emerging HPC architectures, focusing on applications from combinatorial computing and computational science. Due to fundamental physical limitations and power constraints, we are witnessing a radical change in commodity microprocessor architectures to multicore designs. Next-generation petascale and exascale supercomputers will also use multi- and manycore processors as building blocks. Following the successes of accelerators such as the Cell/B.E. processor, general-purpose GPUs and FPGAs, the idea of heterogeneity and specialization in processing cores is also gaining acceptance. There is thus a growing need for novel algorithmic approaches that exploit the performance potential of these emerging multicore platforms, for a wide range of application domains. In addition to algorithm design, I am interested in several related aspects of multicore computing, ranging from programming frameworks to performance analysis and benchmarking.

Multicore Programming: SWARM. SWARM (SoftWare and Algorithms for Running on Multicore) [2] is a portable, open-source framework for designing parallel algorithms and productive programming of multicore systems. SWARM provides constructs for parallelization, restricting control of threads, allocation and deallocation of shared memory, and communication primitives for synchronization, replication and broadcast. Using this framework, we have implemented efficient parallel algorithms for important primitive operations such as prefix-sums, pointer-jumping, symmetry breaking, and list ranking; for combinatorial problems such as sorting and selection; for parallel graph theoretic algorithms such as spanning tree, minimum spanning tree, graph decomposition, and tree contraction; and for computational genomics applications such as maximum parsimony. My primary contributions to this project are the optimization of primitives and algorithms that use the SWARM multi-core framework, and the design of a theoretical model for analyzing algorithmic complexity on homogeneous multicore systems.

Emerging Accelerator Technologies: The Cell/B.E. processor. The heterogeneous multi-core design, asynchronous load/store capability, and the high peak theoretical floating-point performance offered by the Cell processor distinguish it from current general-purpose microprocessors. We have been involved in several projects related to early performance characterization and algorithm design on the Cell processor. We designed an efficient parallel approach on Cell for list ranking, a representative problem from the class of combinatorial and graph-theoretic applications. Due to its highly irregular memory patterns, list ranking is a particularly challenging problem to

parallelize on current cache-based and distributed memory architectures. We describe a generic work-partitioning technique on the Cell to hide memory access latency, and achieve a substantial speedup (up to five times) in comparison to traditional cache-based microprocessors [1]. Our work is one of the first efforts to demonstrate that the Cell processor can accelerate applications outside the multimedia and gaming domains. In ongoing work, we are optimizing the NAS parallel benchmarks, representative of computational fluid dynamics and aerospace applications critical to the NASA mission, on the Cell processor. With Cell-based hybrid supercomputers soon in the offing, this is a timely study on characterizing performance of scientific kernels on Cell.

A Graph Analysis Benchmark for Emerging Architectures. I co-organized an effort to design and implement a parallel graph theory benchmark (hosted at GraphAnalysis.org), that is representative of four key small-world network analysis routines [4]. An early version of this benchmark was part of the DARPA High Productivity Computing Systems (HPCS) Compact Application (SSCA) suite. The benchmark uses a new measure called TrEPS (Traversed Edges Per Second) to compare and characterize performance of current HPC systems on graph-theoretic problems. We have optimized specific kernels of this graph benchmark on various high-end systems such as the IBM BlueGene/L, the SGI Altix 3700 (NASA *Columbia* system), the Sun Niagara processor, and the Cell processor. This effort has been instrumental in identifying bottlenecks to performance of graph-theoretic applications on these novel architectures.

3. Future Research

My long-term research goal is to enable massive-scale information network analysis using novel algorithms and high-performance computing approaches. The design of novel centrality, connectivity, and clustering algorithms, and the SNAP graph framework all represent progress in that direction. This is an exciting time for research in informatics, as we have the perfect storm of data availability and powerful, new architectures.

Dynamic Networks. In real systems, graph abstractions are seldom static: data is dynamically generated and may be assimilated from multiple sources. The analysis of massive transient data streams raises new and complex research problems, and studying static snapshots of the network may be insufficient. We have preliminary results on parallel algorithms and data structures for the topological analysis of dynamic networks [8], and are working on extensions to the SNAP framework for computational support of dynamic graph kernels. Advanced graph analysis queries and kernels such as similarity searches, path-based centrality, and implicit community detection in dynamically evolving data, constitute exciting problems for future research.

Interactome Analysis. The field of systems biology (in particular, the analysis of interactome networks) presents a rich collection of emerging, unsolved combinatorial problems. For instance, one high-level goal is to understand how the global and local properties of complex macromolecular networks impact biological function and observable properties, and how changes in such properties can lead to human diseases. In prior work, we conducted a topological analysis of a large-scale curated human protein-interaction data set [7] to assess lethality and centrality in protein interaction networks. We determined that proteins with high centrality, but low connectivity are abundant in the human interactome network. However, this finding could not be explained by accepted synthetic models for interaction networks. In future research, we intend to extend the interactome analysis for protein function prediction and disease modeling, and also support computational problems arising in interactome analysis using software abstractions implemented in SNAP.

Selected Publications

- [1] D.A. Bader, V. Agarwal, and K. Madduri. On the design and analysis of irregular algorithms on the cell processor: A case study on list ranking. In *Proc. Int'l Parallel and Distributed Processing Symp. (IPDPS 2007)*, Long Beach, CA, March 2007.
- [2] D.A. Bader, V. Kanade, and K. Madduri. SWARM: A parallel programming framework for multicore processors. In *Proc. Workshop on Multithreaded Architectures and Applications (MTAAP)*, Long Beach, CA, March 2007.
- [3] D.A. Bader, S. Kintali, K. Madduri, and M. Mihail. Approximating betweenness centrality. In *Proc. 5th Workshop on Algorithms and Models for the Web-Graph (WAW2007)*, Lecture Notes in Computer Science, San Diego, CA, December 2007. Springer-Verlag.
- [4] D.A. Bader and K. Madduri. Design and implementation of the HPCS graph analysis benchmark on symmetric multiprocessors. In *Proc. 12th Int'l Conf. on High Performance Computing (HiPC 2005)*, Goa, India, December 2005. Springer-Verlag.
- [5] D.A. Bader and K. Madduri. Designing multithreaded algorithms for breadth-first search and st-connectivity on the Cray MTA-2. In *Proc. 35th Int'l Conf. on Parallel Processing (ICPP)*, Columbus, OH, August 2006. IEEE Computer Society.
- [6] D.A. Bader and K. Madduri. Parallel algorithms for evaluating centrality indices in real-world networks. In *Proc. 35th Int'l Conf. on Parallel Processing (ICPP)*, Columbus, OH, August 2006. IEEE Computer Society.
- [7] D.A. Bader and K. Madduri. A graph-theoretic analysis of the human protein interaction network using multicore parallel algorithms. In *Proc. 6th Workshop on High Performance Computational Biology (HiCOMB 2007)*, Long Beach, CA, March 2007.
- [8] D.A. Bader and K. Madduri. High-performance combinatorial techniques for analyzing massive dynamic interaction networks. In *DIMACS/DyDAn Workshop on Computational Methods for Dynamic Interaction Networks*, Piscataway, NJ, September 2007. Rutgers University.
- [9] D.A. Bader and K. Madduri. SNAP, Small-world Network Analysis and Partitioning: an open-source parallel graph framework for the exploration of large-scale networks. In *Proc. Int'l Parallel and Distributed Processing Symp. (IPDPS 2008)*, Miami, FL, April 2008.
- [10] K. Madduri, D.A. Bader, J.W. Berry, and J.R. Crobak. An experimental study of a parallel shortest path algorithm for solving large-scale graph instances. In *Proc. The 9th Workshop on Algorithm Engg. and Experiments*, New Orleans, LA, January 2007.