

# Mobile Grid

Major Area Examination

BY YE WEN

Department of Computer Science  
University of California  
Santa Barbara, CA 93106  
*wenye@cs.ucsb.edu*

## 1 Introduction

Modern computer applications are increasingly concerned with utilizing distributed resources, like processors and storage. Developers of these applications often assume either a homogeneous or a centralized environment without extensive requirement for reliability, security and scalability. This assumption is insufficient for contemporary computation environment, which is essentially heterogeneous, unreliable, exposed to open and insecure world and with dramatically increased entities.

Grid Computing emerges as a technology for coordinated large-scale resource sharing and problem solving among many autonomous groups (referred as *virtual organizations*, VOs)[7]. In Grid's resource model, the resource sharing relationships among VOs are dynamic: VOs can join or leave multiple relationships at any time. However, Grid requires a stable quality of service provided by VOs and the changing of sharing relationship can never happen frequently. This model works for a conventional distributed environment but is challenged in the highly variational wireless mobile environment[30][29][6].

In mobile environment, networks are unstable: mobile nodes can join and leave frequently; the quality of connection is unpredictable; even network itself can be ad-hoc. Moreover, mobile nodes have limited local resources and battery life. Although Grid Computing hasn't addressed these problems extensively, researches in Mobile Computing provide solutions attacking these mobility issues.

It is a natural idea to extend the Grid's resource model to wireless mobile network. However, a lot of questions will arise when mobility is incorporated into Grid. First, we need to justify the integration of Grid and mobile network. Let's look at some scenarios.

**Scientific Application** A biologist is conducting a research on the structure of a protein molecule. He runs a complex protein folding program on a federation of supercomputers which are located among different organizations. To reduce the time complexity of the program, instantaneous interference from the owner of the program may help. The biologist simply carries a *personal digital assistant* (PDA), which will automatically connect to the program wherever and whenever there is a network connection available, to check the progress of protein folding and inject his instructions to direct the running of the program. Important milestones during running and special events need manual manipulations will also be sent to owner's PDA whenever possible.

**Public Service** A forest fire breaks out during hot summer. Firemen equipped with mobile devices are sent out to different locations of the fire spot. The data of the fire are reported through wireless connection to some control center. Furthermore, these data are organized and fed into a simulation program, which takes the current fire input and history information from some database of government institution and runs on the computers of some research centers. The simulation results are sent back to every mobile users as the forecast of the fire's spreading.

**Commercial Business** A project manager of an aeroplane manufacturer flies to a customer's city to demonstrate the design progress of the plane. At the airport, she opens her laptop and connect back to her office via the wireless access point in airport lounge. She finds a new modification to some parts' design on the shared console of her *personal information manager* (PIM). So she initiates the virtual assembly program, which runs over the network across multiple locations of the company, to detect the conflict among different parts. Fortunately, after some manual adjusting, the result is completed just before the boarding time. She doesn't have to worry about it during the flight.

In these scenarios, the application infrastructure has two parts: a static part and a mobile part. For example, in the Scientific Application case, the protein folding program along with other communication and control services are the static part while the PDA that is carried by the biologist is the mobile part. On the other hand, both static part and mobile part are indispensable for Grid applications. Mobile part acts as initiator, monitor, sensor, receiver and director of the application, whose performance greatly affects the performance of the other. For example, what will happen if the biologist can not give his instruction to the protein folding program in time due to a congestion of wireless traffic in the scientific application case, or if the manager can't finish the assembly program before her boarding time in the commercial business case? All these justify the integration of mobility and Grid. We could call it "Mobile Grid".

Then what's the difficulties of this integration? The ultimate target of Grid Computing is performance. However, when mobility is taken into account, the definition of performance can not be limited as computational performance, but incorporates deeper meanings, due to the mobility constraints. We discuss the problems of "Mobile Grid" and relevant solution technologies in the following.

In a Mobile Grid, programs are expected to run efficiently both in computational performance and energy. The necessity of energy efficiency is due to the limited lifetime of batteries for mobile devices. The programs running on them should minimize power consumption to achieve the largest availability. Conventional resource management and scheduling algorithms are applicable in this area to achieve computational performance. For energy efficiency, either the scheduling algorithm can be extended or special mechanisms are needed, for example, voltage scaling.

Just like mobile applications, an application in a Mobile Grid needs adaptability. In the mobile environment, unpredictable variations of network connections require the application to be adaptable and responsive to the changes and adjust its behavior correspondingly to provide a graceful QoS. The adaptability actually also improves both the computational and energy efficiency. Various technologies are proposed to achieve the adaptability, such as caching and proxy.

Security problem is exaggerated in Mobile Grid. One reason is due to the intrinsic vulnerability of wireless network. The other is that security imposes bigger performance impact on the mobile devices than that on a normal computer. The encryption/authentication mechanisms in current Grid implementation may be too clumsy and power-consuming for a small PDA. Security problems should not be separated from the consideration of performance and energy efficiency.

Reliability also has an extended meaning in Mobile Grid. For example, network disconnections a normal phenomenon for a mobile device. Application should be aware of it and handle gracefully, not just regarding it as an exception. Adaptation mechanism can be used to deal with the new reliability model. Peer-to-Peer (P2P) computing technologies also provide interesting features for robust distributed storage.

There are also other concerns, like address mobility and location independent naming, which are studied in the area of mobile network researches, and scalability problem, for which P2P presents a good model.

In the following sections, I summarize important research works of above topics in more detail. Since each topic is a big area, only those most relevant to Mobile Grid will be discussed.

## 2 Overview of Grid Computing

Grid problem is defined as coordinated resource sharing and problem solving in dynamic, multi-institutional virtual organizations. The central problem here is the interoperability. A Grid system must provide a set of standard interfaces for applications to access all the services VOs provide.

*Open Grid Services Architecture* (OGSA)[8] is an open standard of Grid architecture based on the work of Globus. OGSA adopts a uniform service-oriented model, which takes every resource entities and programs as services (called *Grid service* specifically). Also, services are defined abstractly using WSDL to make them independent of actual implementation and platform. To flexibly manipulate services, a set of basic OGSA interfaces is defined as the convention about how services are interacted with. These includes the interfaces to create services (*Factory*), publish services (*Registry*), discover/terminate services (*GridService*), manage service handles and references (*HandleMap*), and handle notification messages (*NotificationSource/NotificationSink*). With these interfaces, all Grid services can be constructed in a uniform way. To make architecture general, other important issues, such as reliability and authentication, are considered as service protocol bindings, which are external to the core service set, but should be addressed in a complete implementation.

## 3 Scheduling

Scheduling is the key to the resource management problem. It essentially can be considered as a maximization problem with resource status and application data as the parameters. There are great amount of work on scheduling in distributed computing area.

*Application-level scheduling* (AppLeS)[1] addresses the scheduling in a Grid environment. In contrast with other general scheduling methods, AppLeS is application-specific. It aims to gather and predict dynamic information from system and application and utilize them to evaluate the scheduling plan from the point of view of the application. AppLeS has a generalized organization that can be applied to build scheduler for resource intensive applications. The execution part of AppLeS, the *coordinator*, collects a set of resource sets from *resource selector* and a set of planners from *planner*, evaluates the performance using *estimator* and activate the optimized plan with *actuator*. The sources of information are important in AppLeS. Network Weather Service is used to monitor and predict resource load. Application specific information comes from user specification and application models.

When mobility is taken into account, special scheduling algorithm is of interest. In a mobile environment, computational resources are normally composed by a set of powerful static hosts and a larger set of less powerful mobile hosts. This can be regarded as a host-satellite structure, for which there are available algorithms to solve the basic problem. The simplest case of the problem is composed by two non-identical processors and a serial program with  $N$  components. In [34] the structure of the program is modeled as an intermodule-connection graph and extended with two extra nodes for two processors. Then network flow algorithm is used to find the min-cut set, which corresponds to the scheduling solution. [2] provides the algorithm for a more complex host-satellite system: single host, multiple satellites. Method in [34] is used to convert the programs into chain-like structures. A layered assignment graph is then built to model the scheduling problem. And the algorithm of finding optimal sum-bottleneck path is used to find the schedule solution.

## 4 Energy-aware Computing

In mobile environment, performance also concerns energy. Mobile devices use battery as its power supply and battery has a limited life time. So to make applications energy-efficient becomes a big issue in mobile computing.

Many components on mobile devices have the ability to work in different performance levels, which in turn cost different energy. Energy can be saved by making a trade-off between energy consumption and performance. This is called dynamic power management, including voltage-scaling[23], disk spin-down and network interface hibernating etc. Paleologo[17] proposed a method to find optimal policy which maximizes the average performance level of a system using stochastic optimization. In this method, system components (service provider), user (service requestor) and task queue (service queue) are modeled by discrete time Markov decision chains. Then average performance/penalty is composed from the models and maximized/minimized by solving the Markov decision process optimization problem. [24] and [32] improved Paleologo's method by using more advanced Markov decision process models.

Another way to save mobile devices' power is by offloading some part of the computation to a static infrastructure via a wireless connection. A partition scheme about how to partition the tasks within a program is the key to optimize the performance of the program. Li[13] provided a method to make this optimization of partition. Li's method consider each function call site as a task. A program is parsed and a cost graph is built for all its tasks and data. The cost graph specifies the cost of energy consumption and the penalty of communication. Then the total energy consumption is composed from the cost graph. Heuristic branch and bound algorithm is used to find the optimized solution of this problem. [28] verifies that by remote execution, offloading can save energy.

[12] and [3] studied the energy saving problem from the view of compiler.

## 5 Adaptation

Mobile environment is highly variational. Application must be always aware of the change of environment and take prompt reaction in order to offer graceful QoS.

Disconnection is a very common situation for wireless network and mobile hosts. Disconnected operation is a mode of operation that enables a client to continue accessing critical data during temporary failure of the network. *Coda* file system[11][31] provides a good example to illustrate the effectiveness of disconnected operation. The idea of Coda is simple: a file system proxy, Venus, which resides on mobile client, hoards the objects that applications are using or going to use. When client is disconnected from file server, Venus acts as the real server to provide transparent data access from its local cache. Upon reconnection, local operations are integrated with the master copies on server. Many associated critical problems, like cache consistency and high availability, are solved during the lifetime of Coda's development, which makes it a successful system. [35] solve the disconnected problem by undirected message passing.

More common change of network is the degrade of its quality. Applications need to adapt to this change agilely in their own ways. *Odyssey*[16] provides the architecture for agile application-aware adaptation. Odyssey adapts to the environment by changing the fidelity of data, which means the degree to which data presented at a client matches the copy at server. Odyssey runtime consists of a viceroy and a set of wardens. Each warden deals with a specific type of data. Applications access data via wardens. Viceroy manages system resources centrally. Odyssey is implemented by intercepting system calls in NetBSD system kernel and transferring the request to Odyssey runtime. When changes happen, Odyssey runtime invokes callbacks specified by the request, which in turn, changes the fidelity of data to adjust to the environment.

There are other adaptive systems. *Conductor*[36] proposed a distributed adaptation system. *QEX*[4] uses RPC-based method. *Limbo*[5] is based on tuple space.

## 6 Mobile IP & Ad-hoc Networks

Mobile networks can be classified into two classes: nomadic network and ad-hoc network. Nomadic network is composed by a static infrastructure and a set of mobile nodes. Mobile nodes communicate via the accessing of static network. Ad-hoc network is purely composed by mobile hosts. There is no static infrastructure support.

In a nomadic network, the problem is the seamless migration of mobile nodes among static networks. *Mobile IP*[21][20] is an IP layer (Data Link layer) protocol to provide local independent addressing for mobile hosts. It is composed by a home agent, the home network, a foreign agent, the foreign network and the mobile node. The mobile node associates with two IP addresses: the permanent home address and the temporary care-of address, which is used in the foreign network. When the mobile node moves from its home network to the foreign network, it obtains the care-of address from foreign network and registers it at the home agent. Then the home agent intercepts traffic to the mobile node and redirect it to the mobile node by tunneling or to the foreign agent in the foreign network, which relays the traffic to the mobile node. Mobile IP has been incorporated into the next generation of IP protocol, IPv6.

In ad-hoc networks, mobile nodes only have knowledge of their neighbor nodes. So how to route the packets to other nodes is a big problem. There are many ad-hoc routing protocols[27], like DSDV[18]. *Ad-hoc On-Demand Distance Vector Routing (AODV)*[19] aims to achieve a scalable ad-hoc network by minimizing the routing traffic. Like many other routing protocols, AODV maintains a routing table for each mobile nodes, which contains the active routes and the metrics. AODV initiates a path discovery process when a new route is needed and it is not in the routing table by broadcasting a request to its neighbors. Its neighbors in turn rebroadcast the requests until the destination is reached. The reply then returns to the source node following the reverse link keeping at each intermediate node. Sequence numbers and timeouts are used to prevent the request repetition and ensure the freshness of route.

## 7 Peer-to-Peer Computing

Peer-to-Peer (P2P)[14] computing provides many useful technologies in building a Mobile Grid, such as its reliability and scalability model. One of the major challenges for P2P computing is resource locating and routing problem. One can imagine keeping all other nodes addresses for each node in the network. The locating and routing can be achieved in one hop. But this scheme does not scale. A good protocol should be both reliable and scalable.

There are several object location and routing schemes. They are essentially distributed hash tables over large-scale networks, like *CAN*[25], *Chord*[33] and *Tapestry*[10]. Pastry[26] is one that loosely bases on Plaxton-tree structure[22]. The basic idea is to embed many trees into the network. Each node itself is the root of one of these trees. In Pastry, each node is assigned a random ID. The tree is then built for each node according to the similarity of IDs. Documents stored in the network are also assigned random IDs. A document are kept on the node with the most similar ID and also cached along the route from search initiator to destination to increase availability and reliability. Each node only keeps  $O(B \cdot \log_b N)$  states and routing can be completed in average of  $O(\log_B N)$  hops. Pastry is also enhanced by other important features like dynamic handling of node arrival and departure, adaptation to failure and locality.

## 8 Security

Due to the resource limitation of mobile devices, security for Mobile Grid is difficult, even without the inherent risks posed by wireless media. Performance issue hinders the deploying the authentication mechanisms used in common Grid architecture on mobile hosts.

*Charon*[9] builds a Kerberos[15] authentication mechanism for mobile devices. It partitions the Kerberos client functionality into a simple client module that does little more than DES encryption and a proxy module running the remainder of Kerberos client protocol. The authentication process is in two phases. Client first establishes a secure communication channel with proxy by considering proxy as a Kerberos service. Then via this channel, proxy talks with other Kerberos service on behalf of client and relay the messages between client and other services. Charon's proxy never has enough information to talk with services on behalf of client without client's cooperation. This ensures that the security of the communication via proxy will not be compromised. Meanwhile, client has very little burden of computation which makes it suitable to run on a resource-limited mobile device.

## Bibliography

- [1] F. Berman, R. Wolski, S. Figueira, J. Schopf, and G. Shao. Application-Level Scheduling on Distributed Heterogeneous Networks. *In Proceedings of Supercomputing*, November 1996. Pittsburgh, PA, USA.
- [2] Shahid H. Bokhari. Partitioning problems in parallel, pipelined and distributed computing. *IEEE Transactions on Computers*, 37(1):48–57, 1988.
- [3] Eui-Young Chung, Luca Benini, and Giovanni De Micheli. Source code transformation based on software cost analysis. *International Symposium on Systems Synthesis (ISSS2001)*, 2001. Montreal, Quebec, Canada.
- [4] Nigel Davies, Adrian Friday, Gordon Blair, and Keith Cheverst. Distributed systems support for adaptive mobile applications. *ACM Mobile Networks and Applications, Special Issue on Mobile Computing - System Services*, 1(4), 1996.
- [5] Nigel Davies, Stephen Wade, Adrian Friday, and Gordon Blair. Limbo: A tuple space based platform for adaptive mobile applications. *In Proceedings of the International Conference on Open Distributed Processing/Distributed Platforms (ICODP/ICDP '97)*, pages 27–30, May 1997. Toronto, Canada.
- [6] Dan Duchamp. Issues in wireless mobile computing. *In Proceedings of the third IEEE Workshop on Workstation Operating Systems*, April 1992. Key Biscayne, Florida.
- [7] I. Foster and C. Kesselman. The anatomy of the grid: Enabling scalable virtual organizations. *International Journal on Supercomputing Applications*, 15(3), 2001.
- [8] Ian Foster, Carl Kesselman, Jeffrey M. Nick, and Steven Tuecke. The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration. Open Grid Service Infrastructure WG, Global Grid Forum, June 2002.
- [9] Armando Fox and Steven D. Gribble. Security on the move: indirect authentication using kerberos. *In Proceedings of the annual international conference on Mobile computing and networking*, 1996. Rye, New York, USA.
- [10] Kirsten Hildrum, John D. Kubiawicz, Satish Rao, and Ben Y. Zhao. Distributed object location in a dynamic network. *In Proceedings of the Fourteenth ACM Symposium on Parallel Algorithms and Architectures (SPAA)*, 2002.
- [11] James J. Kistler and M. Satyanarayanan. Disconnected operation in the coda file system. *ACM Transactions on Computer Systems*, 10(1):3–25, February 1992. Atlanta, Georgia, USA.
- [12] U. Kremer, J. Hicks, and J. Rehg. A compilation framework for power and energy management on mobile computers. *In Proceedings of the 14th International Workshop on Parallel Computing (LCPC'01)*, August 2001.
- [13] Zhiyuan Li, Cheng Wang, and Rong Xu. Computation offloading to save energy on handheld devices: a partition scheme. *In Proceedings of the ACM international conference on compilers, architecture, and synthesis for embedded systems*, 2001. Atlanta, Georgia, USA.
- [14] Dejan S. Milojevic, Vana Kalogeraki, Rajan Lukose, Kiran Nagaraja1, Jim Pruyne, Bruno Richard, Sami Rollins, and Zhichen Xu. Peer-to-peer computing. *HP Labs Technical Report, HPL-2002-57*, 2002.
- [15] B. Clifford Neuman and Theodore Tso. Kerberos: An authentication service for computer networks. *IEEE Communications*, 32(9), September 1994.
- [16] Brian D. Noble, M. Satyanarayanan, Dushyanth Narayanan, James Eric Tilton, Jason Flinn, and Kevin R. Walker. Agile application-aware adaptation for mobility. *In Proceedings of the ACM Symposium on Operating System Principles*, October 1997. St. Malo, France.
- [17] G. A. Paleologo, L. Benini, A. Bogliolo, and G. De Micheli. Policy optimization for dynamic power management. *In Proceedings of Design Automation Conference*, pages 182–187, June 1998.
- [18] C. E. Perkins and P. Bhagwat. Highly dynamic destination-sequenced distance-vector routing (dsv) for mobile computers. *In Proceedings of the ACM SIGCOMM conference on Communications architectures, protocols and applications (SIGCOMM'94)*, 1994.
- [19] C. E. Perkins and E. M. Royer. Ad-hoc On-Demand Distance Vector Routing. *In Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications*, pages 90–100, February 1999. New Orleans, LA, USA.
- [20] Charles E. Perkins. Ip mobility support. *RFC 2002*.
- [21] Charles E. Perkins. Mobile IP. *IEEE Communications Magazin*, May 1997.
- [22] C. G. Plaxton, R. Rajaraman, and A. W. Richa. Accessing nearby copies of replicated objects in a distributed environment. *Theory of Computing Systems*, 32:241–280, 1999.
- [23] Johan Pouwelse, Koen Langendoen, and Henk Sips. Dynamic voltage scaling on a low-power microprocessor. *In Proceedings of the seventh ACM annual international conference on Mobile computing and networking*, 2001. Rome, Italy.

- [24] Q. Qiu and M. Pedram. Dynamic power management based on continuous-time markov decision processes. *Design Automation Conference*, pages 555–561, 1999.
- [25] Sylvia Ratnasamy, Paul Francis, Mark Handley, Richard Karp, and Scott Shenker. A scalable content-addressable network. *In Proceedings of the SIGCOMM 2001*, pages 161–172, 2001.
- [26] A. Rowstron and P. Druschel. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. *IFIP/ACM International Conference on Distributed Systems Platforms (Middleware)*, pages 329–350, November 2001. Heidelberg, Germany.
- [27] Elizabeth M. Royer and C.-K. Toh. Review of current routing protocols for ad-hoc mobile wireless networks. *IEEE Personal Communications Magazine*, April 1999.
- [28] Alexey Rudenko, Peter Reiher, Gerald J. Popek, and Geoffrey H. Kuenning. Saving portable computer battery power through remote process execution. *ACM Mobile Computing and Communications Review*, 2(1), 1998.
- [29] M. Satyanarayanan. Fundamental challenges in mobile computing. *In Proceedings of the fifteenth annual ACM Symposium on Principles of Distributed Computing*, 1996. Philadelphia, Pennsylvania, USA.
- [30] M. Satyanarayanan. Pervasive computing: Vision and challenges. *IEEE Personal Communications*, August 2001.
- [31] M. Satyanarayanan. The evolution of coda. *ACM Transactions on Computer Systems (TOCS)*, 20(2), 2002.
- [32] T. Simunic, L. Benini, P. Glynn, and G. De Micheli. Event-driven power management. *IEEE Transactions on Computer-Aided Design*, July 2001.
- [33] Ion Stoica, Robert Morris, David Liben-Nowell, David R. Karger, M. Frans Kaashoek, Frank Dabek, and Hari Balakrishnan. Chord: A scalable peer-to-peer lookup protocol for internet applications. *To Appear in IEEE/ACM Transactions on Networking (after 2002)*.
- [34] Harold S. Stone. Multiprocessor scheduling with the aid of network flow algorithms. *IEEE Transactions on Software Engineering*, SE-3(1), January 1977.
- [35] Peter Sutton, Rhys Arkins, and Bill Segall. Supporting disconnectedness - transparent information delivery for mobile and invisible computing. *IEEE International Symposium on Cluster Computing and the Grid (CCGrid 2001)*, pages 15–18, May 2001. Brisbane, Australia.
- [36] Mark Yarvis, Peter Reiher, Kevin Eustice, and Gerald J. Popek. Conductor: Enabling distributed adaptation. *UCLA Tech Report CSD-TR-010025*, June 2001.