

A Collaborative Problem-Solving Framework for Mobile Devices

Stan Kurkovsky
Columbus State University
4225 University Ave.
Columbus, GA 31907
1-706-565-3520

Kurkovsky_Stan@colstate.edu

Bhagyavati
Columbus State University
4225 University Ave.
Columbus, GA 31907
1-706-565-3520

Bhagyavati@colstate.edu

Arris Ray
Columbus State University
4225 University Ave.
Columbus, GA 31907
1-706-568-2410

Ray_Arris@colstate.edu

ABSTRACT

The explosive growth in the number of mobile devices such as Internet-enabled cellular phones, wireless handheld devices, wireless laptops, and tablet PCs has driven the corresponding growth in applications for mobile computing. These applications usually belong to one of two classes: collaborative applications and individual application. While collaborative applications require several mobile devices to work together and include peer-to-peer computing and grid computing, individual applications are local to the mobile device and its user. In this paper, we present a framework that allows mobile devices to collaboratively work on a computationally-expensive problem. Such a problem is decomposed into smaller tasks and distributed across other mobile devices willing to share their computational power with others. This framework is based on the paradigm of grid computing applied to the domain of wireless mobile devices. This paper presents our current implementation of the framework architecture and simulation of its functionality. Also, we focus on the issues of mobility, QoS and network stability and their effect on the performance of our collaborative problem-solving framework.

Categories and Subject Descriptors

C.1.3 [Processor Architectures]: Other Architecture Styles – *cellular architecture*;

C.2.1 [Computer Networks]: Wireless communication;

C.2.4 [Computer Networks]: Distributed systems – *distributed applications*.

General Terms

Design, Experimentation.

Keywords

Grid architecture, wireless network, mobile devices.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ACMSE '04, April 2-3, 2004, Huntsville, Alabama, USA.

Copyright 2004 ACM 1-58113-870-9/04/04...\$5.00.

1. INTRODUCTION

Mobile computing is an emerging area of modern application-driven research and is important for many reasons. One of these reasons is the explosive growth of the number of devices available on the market and owned by customers. These devices include Internet-enabled cellular phones, wireless PDA's, wireless laptops, and the most recent addition to this class of devices – tablet PCs. There are several types of computing applications for mobile devices. They can be broadly divided into two classes: collaborative applications such as peer-to-peer (P2P) and grid computing, and individual applications local to the mobile device and its user [3,9,10]. Collaborative applications, by definition, require collaboration between or distribution of the computational load across two or more mobile devices. Individual applications, on the other hand, are initiated and executed completely on a single device.

Especially for the first class of applications that embraces collaborative applications and/or resource sharing, the mobility of the devices is a critical factor affecting their performance [8]. In other words, if the devices participating in a collaborative task are highly mobile, this may adversely impact the time taken to perform the computationally-intensive task.

The issue of mobility is interconnected to the issue of Quality of Service (QoS) [12]. The long-term goal of the collaborative applications for mobile wireless devices is to provide the same QoS to mobile users as experienced by wired users. There is a definite trend in mobile applications that recently was available only to the users of desktops. For example, instant messaging is one of the widely used wireless applications; however, long before it became available to the wireless users, it was pioneered on desktops with such applications as ICQ. File-sharing applications constitute another example of applications that were, in the recent past, impractical to execute on mobile devices due to the lack of processing power and bandwidth; they are now becoming mainstream applications on mobile devices. In these applications, soft handoffs, wherein device migration is handled transparently to the user, aids in improving wireless QoS. Due to the inherent characteristics present in wireless networks, frequent disconnection may result in poor QoS.

The two major factors that cause network instability are the nature of the wireless medium and the mobility of the devices [11]. Interference can be caused in the wireless channels of

communication due to atmospheric disturbances such as electromagnetic storms or environmental factors such as rain. The mobility of the devices involved in the wireless applications can cause network instability. Load balancing techniques employed in wired networks can help mitigate such imbalances

and restore network stability.

Since convenience is a predominant factor in the growth of mobile devices, it has driven the characteristics of the devices, such as screen size, overall size, processing power and battery power. Mobile devices are dependent on battery power for their

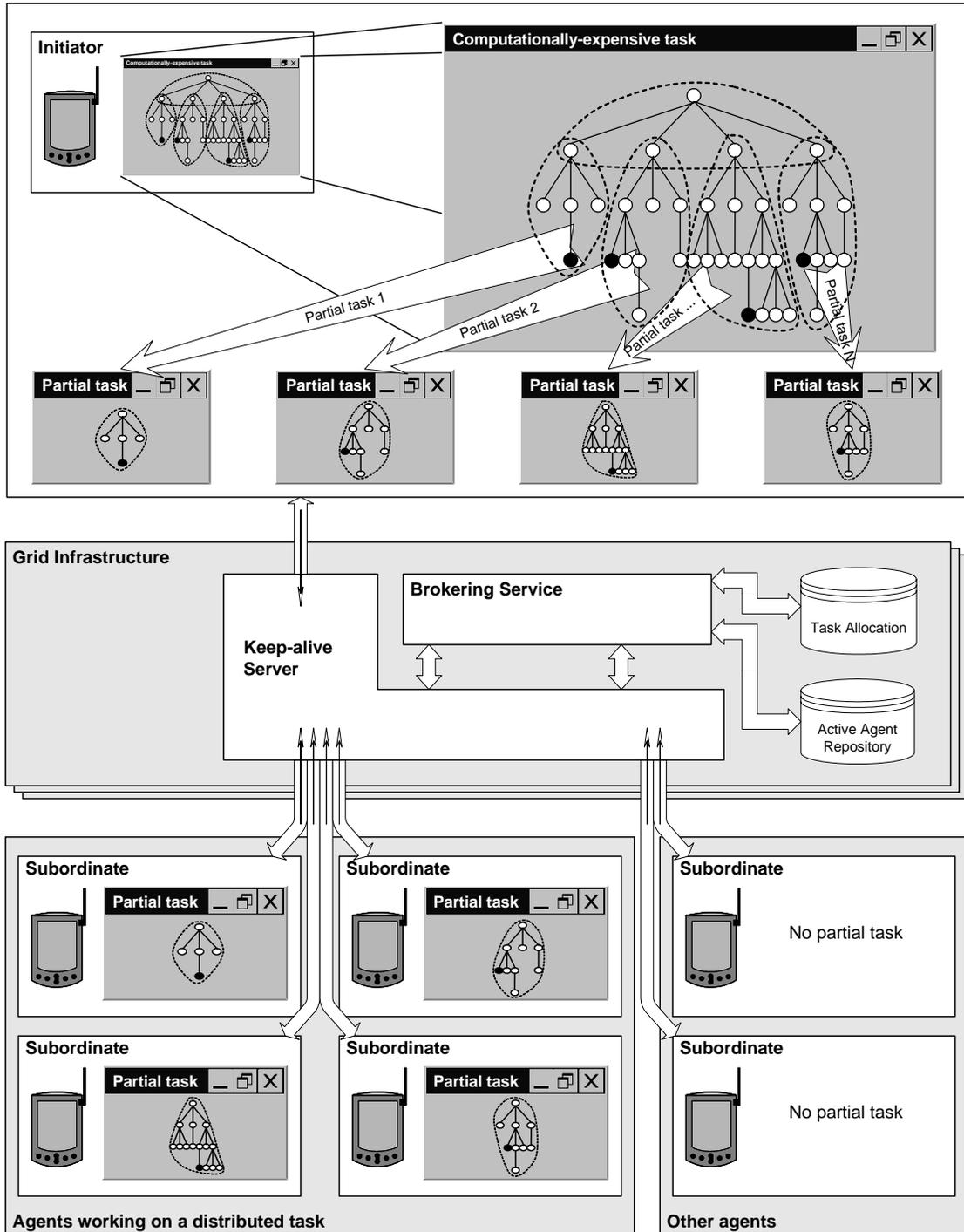


Figure 1. Grid-Based Collaborative Framework Architecture for Mobile Devices

functioning [7]. On the one hand, mobile users demand longer lasting batteries; on the other, they do not want to carry bulky devices or spare batteries. When a mobile device is connected to the network, it uses more power than when it is not connected to the network. Devices on standby mode consume more power than in the turned-off mode. The design of the wireless applications for mobile devices assumes that the devices have sufficient power to accomplish their tasks. With advances being made in prolonging the longevity of batteries, we assume that power dependency is not as critical a factor as mobility and the resultant network instability and QoS issues. Furthermore, the miniaturization of device components implies that more space is available for primary and backup batteries.

In this paper we present a grid-based design for a collaborative problem-solving framework for wireless networks, which serves mobile devices with limited processing power [1]. Here, we focus on the issues of mobility, instability, handoff and QoS in collaborative wireless applications running in our framework. The paper is organized as follows. Section 2 describes concept of the grid-based collaborative problem-solving framework. Section 3 addresses major issues in our work, while Section 4 elaborates on the design and implementation for the proposed collaborative framework. Sections 4 and 5 explain the simulation scenarios and a simulation of the framework in depth. Section 6 presents the conclusions thus far, and the work to be conducted in the near future. References to related literature are provided in Section 8.

2. PREVIOUS WORK

In our previous work, we developed an architecture enabling mobile devices within a wireless cell to form a computational grid in order to collaboratively solve resource-intensive tasks that they would not be able to complete on their own (Figure 1). In our approach, we modeled each mobile device after an autonomous intelligent agent [4,6] that facilitates solving a user-requested task in such a way that the user is unaware of the exact method that it is using to solve the task. Depending on the nature and computational complexity of the task, an agent may be capable of solving it on its own, or, when the task is too big and/or resource-intensive, the agent may decide to seek collaboration from the grid infrastructure. The grid framework may offer help on solving such resource-intensive tasks by distributing it across the community of other agents connected to the grid and facilitating the communication and collaboration among these agents.

In order for a mobile device to be able to take advantage of the computational benefits offered by the grid, it must run a client application implementing a light weight communication protocol for information exchange with the grid *Brokering Service*. Any device running such a client application is called a *Subordinate* and can be requested to collaborate with other devices by participating in the solution of a part of some computationally-intensive task. Any Subordinate may become an *Initiator* of a distributed task if its user requests a large and/or computationally intensive task to be solved. In this case, the initiator is responsible for submitting such a distributed task to the *Brokering Service*. The *Brokering Service* facilitates all communication among the devices participating in the grid activities.

The *Brokering service* contains two major information stores, *Active Agent Repository (AAR)* and *Task Allocation Table (TAT)*. *AAR* contains information about all computing devices

that are available to the grid, which includes all mobile devices within a given wireless cell. This information includes the CPU rating, amount of available memory and the current level of the battery charge. *TAT* contains information describing how each distributed task is allocated across the agents on the grid. Information stored in *TAT* evolves as the distributed task gets closer to completion. When subordinates complete their partial tasks, they return the results back to the *Brokering Service*, which stores them in *TAT* where they await to be requested back by the *Initiator* of the corresponding task.

All communication aspects of the grid infrastructure are handled by the *Keep-Alive Server*, which sends and receives messages to and from all computing devices available on the grid. All devices entering the wireless cell and willing to participate in the distributed grid tasks advertise themselves as available to the *Keep-Alive Server*. A special mechanism (described in our previous work) is put in place to prevent computing devices from over-using the grid resources without making their own resources available to the rest of the grid [5]. There is a clear separation between the *Keep-Alive Server* and the *Brokering Service* based on their functionality, as illustrated in Table 1.

Table 1. Functional Decomposition of the Main elements of the Wireless Grid Architecture

Framework Element	Main Purpose
Brokering Service	Task distribution
Keep-Alive Server	Communication
Initiator	Task semantics
Subordinates	Processing power

3. MAJOR AREAS OF CONCERN

As we noted previously, the mobility of the devices involved in the grid tasks is a critical factor affecting the performance of the entire grid. If the devices in the grid are highly mobile, this may increase the time taken to perform a computationally-intensive task. A related issue is that of subordinates and/or the initiator leaving the cell while partial tasks are still in progress. Currently, if one or more subordinates involved in working on partial tasks leave the grid, the *Brokering service* re-allocates these partial tasks to other subordinates present in the grid. If we modify this approach to allow the subordinates to carry their partial tasks with them when they leave the grid and migrate to a new service area, then handoff issues become paramount. When the initiator of a task moves to a new area, the *brokering service* is currently programmed to abort all partial tasks associated with the initiator. However, if the grid were to facilitate task completion even when the initiator has migrated to a different cell, handoff issues become a significantly complicating factor. In order to ensure compatibility with today's 3G technologies, soft handoff has to be ensured for initiator and subordinate migration so that the user is unaware of the underlying issues involved with migration.

In a typical wireless application, the issue of Quality of Service (QoS) is strongly related to the mobility of the devices. Soft handoffs, making the cross-cell device migration transparent to end-user, improve the wireless QoS. Therefore, handoff issues

need to be addressed by the wireless grid architecture presented. Frequent disconnections that are inherent to the wireless networks may result in poor QoS. If such abnormal disconnections are considered and mitigated at the architectural design stage, then mobile users can expect better QoS than currently being offered.

The mobility of the devices involved in the wireless grid can cause network instability because the underlying services, such as the Brokering service and the Keep-Alive server, can be quickly overwhelmed by a large number of devices entering the grid at the same time. By the same token, if a large number of subordinates leave the cell at the same time, the network could become unstable due to the lack of sufficient resources to continue working on initiated tasks. To address this issue, load balancing techniques employed in wired networks can be used to resolve such imbalances and restore network stability.

In the following section we describe the design, implementation, and goals of experiments that are primarily aimed at discovering the effects of network instability on the computational grid of mobile devices.

4. WIRELESS GRID SIMULATOR

4.1 Motivation

Grid computing constitutes a relatively new field of research that is primarily characterized by the capability to allow a heterogeneous population of computing devices to share computational resources across the network. This is achieved by means of a network infrastructure which manages the aggregation of these resources and transparently presents them as a uniform commodity to devices seeking to perform a computationally-intensive task beyond their individual capabilities. Although a variety of wired grid infrastructures have been developed, most studies indicate that SOAP-based XML Web Services play a central role in grid middleware, allowing for future grid implementations to build on the TCP/IP foundation of the global Internet [2].

Another notable aspect of current grid implementations is the focus on high-performance computing. A major motivating factor in researching grid networks, particularly in the academic and scientific communities, is the opportunity for massively parallel processing of raw data generated from processor intensive simulations, such as the modeling of global weather patterns or large-scale signal processing. Consequently, such grid populations usually consist of resource-rich, server-class computers; these homogeneous populations obscure critical issues that arise from population heterogeneity, which include the mobility of network devices and the repercussions of mobility on resource sharing.

A grid simulator has been developed for the purpose of experimentally determining the effects of device mobility and resource-scarcity on resource sharing. In our simulations, we restrict the resources available for sharing only to the processing power of network devices. The sharing of resources is emulated by the distribution of a parallel task, which represents some workload that requires more processing power than that available on a single mobile device. Thus, decomposed units of a task are distributed to multiple peer devices for processing, and sub-task results are returned to the initiating device, which may then determine a complete solution based on partial results.

4.2 Architecture

We developed a simulator to effectively determine the result of introducing mobility into a computational grid in lieu of access to an extant grid or a sufficient number of mobile devices available for actual experimentation and testing. A high-level view of the simulator architecture consists of the following three components: 1) client application installed on participating devices, 2) Keep-Alive Server, and 3) Brokering Service.

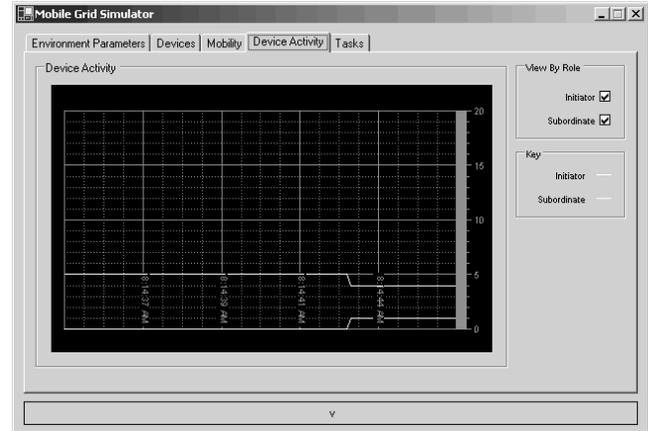


Figure 2. Device Activity (Initiators and Subordinates) Monitored by the Grid Simulator

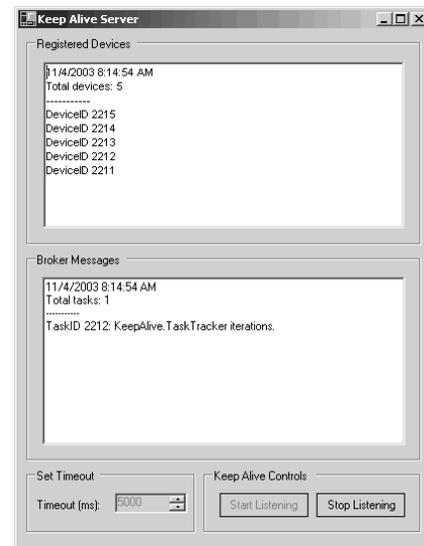


Figure 3. Mobile Device Population Monitored by the Keep-Alive Server

4.2.1 Client Application

A client application is installed on mobile devices which permits access to and communication across the grid. From the simulator's point of view, the client application is synonymous with the mobile device itself. Mobile devices are characterized by the role they play in the grid, as shown in Figure 2. The role of the mobile device is determined by whether it initiates a task or

whether it solves a part of the task of another device. As we noted earlier, if the mobile device initiates a task, it is referred to as the Initiator. If the mobile device participates in solving some other device's task, it is referred to as a Subordinate. Devices access different sets of functionality based on their current role. Subordinates may request to join or leave the grid, initiate a task, run assigned sub-tasks or return completed sub-tasks. Initiators may abort an initiated task or retrieve results for the completed sub-tasks.

4.2.2 Keep-Alive Server

The Keep-Alive Server is a TCP/IP service that is used to aid the Brokering Service in maintaining an accurate record of the state of the grid, as shown in Figure 3. The mobile devices on the grid communicate with the Keep-Alive Server via the keep-alive protocol at periodic intervals to signify that they are still active in the grid. Changes to the state of the grid are reported to the Keep-Alive Server from network devices and the Brokering Service. The time-sensitive nature of the keep-alive protocol requires that network devices communicate directly with the server to avoid timing-out; however, all time-independent changes occur via the Brokering Service, which notifies the Keep-Alive Server of changes made to the state of the grid through the use of custom TCP packets.

4.2.3 Brokering Service

The Brokering Service is a suite of XML Web Services that exposes methods used to record detailed information regarding the grid population as well as device-initiated tasks; this information is maintained in an Active Agent Repository (AAR) and a Task Allocation Table (TAT), respectively. The AAR is used to maintain information related to various characteristics of each device registered with the grid, including the device type, role (Subordinate/Initiator), processor speed and physical memory. The TAT is used to maintain information related to any device-initiated tasks, including a list of arguments required for task execution, the identity of the Initiator and task executable byte-code. Updates made to either the AAR or TAT are propagated via TCP packets sent by the Brokering Service to the Keep-Alive Server, which is kept current of all changes to the state of the grid.

Upon a device's request to join the grid, provided the grid is capable of accommodating the new device, the characteristics of the device are recorded into the AAR. Changes that occur to these characteristics are reported to the Brokering Service via the Keep-Alive protocol, whereupon the AAR is updated with the new information. This ensures that the Brokering Service is capable of efficiently determining an ideal distribution for any initiated tasks.

If a subordinate finds that it is necessary to initiate a task, source code and relevant parameters are submitted directly to the Broker, which dynamically compiles the code into an executable. This data, as well as the ID of the initiator, is recorded into the TAT. Following that, the Broker assigns the newly submitted task to available subordinates. The Keep-Alive Server is then notified that a task is ready for distribution and transmits the task to the appropriate devices, specified by the Brokering Service. Since the work described here presents a simulation of the grid infrastructure, we deliberately made no attempt of certifying the

source code deployed from the initiator as harmless. However, this can be achieved by using digital certificates issued by trusted certifying agencies.

5. SIMULATION SCENARIOS

The grid simulator currently provides and allows modifying three environmental variables: initial population size, device mobility and task initiation frequency (TIF). The initial population size is used to specify the number of devices in the grid at the beginning of a simulation run. Device mobility refers to the probability of a new device joining the grid or a current device leaving the grid per unit of time. The frequency of task initiation refers to the probability that a device will initiate a task per unit of time. The criterion used to gauge the efficiency of the grid is the total time taken for a task to be initiated, distributed, solved and returned to the initiating device. In other words, the time taken to completely execute a resource-intensive task is the metric, which depends on 3 control variables: the grid population, device mobility and TIF.

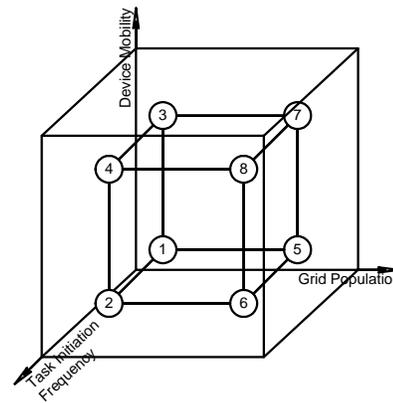


Figure 4. Experimental Space Determined by the Three Control Variables

These three environmental control variables result in twelve distinct scenarios that we plan to execute through trial runs to determine the average task execution time in each case. As a result of these experiments, we can infer the efficiency of the grid architecture as a whole. The cube shown in Figure 4 represents the experimental space yielded by the three environment variables without any specific coordinate values. Instead, the scenario cube merely indicates that each variable will be tested with a range of values between some minimum and maximum. Numbered vertices of this cube represent the ranges of the environment variables. Edges of the cube correspond to the actual scenarios, in which one variable can change its values while the other two remain fixed. For example, the point 8 represents high density of grid population, high device mobility and frequent task initiation in the grid. Trivial values will be excluded in the trials, and the remaining variables will provide the control state in any given scenario.

The results produced by executing the various simulation scenarios described above will be grouped and graphed to produce the 12 distinct graphs enumerated in Table 2.

Table 2. Summary of Experimental Trials

Initial Grid Population	Device Mobility	Task Initiation Frequency
Variable	Low	Low
Variable	Low	High
Variable	High	Low
Variable	High	High
Low	Variable	Low
Low	Variable	High
High	Variable	Low
High	Variable	High
Low	Low	Variable
Low	High	Variable
High	Low	Variable
High	High	Variable

These graphs will be used to isolate and identify direct effects of the three environment variables on the state of the grid. These experimental results will yield insights into the factors causing network instability and ways to minimize its adverse consequences on task completion time.

6. SUMMARY AND FUTURE WORK

Our immediate plans include conducting simulation trials according to the scenarios outlined in the previous section. We also would like to consider a range of diverse applications for the collaborative problem-solving framework.

One possible application for this framework architecture is to enable effective ad-hoc collaboration across mobile devices in emergency situations. Emergency situations may be characterized by a presence of a relatively large number of wireless mobile devices in the absence of a reliable wireless infrastructure. For example, in case of natural disasters, such as earthquakes, most mobile devices will most likely remain intact while wireless transmission towers are very likely to be damaged. There is a clear need to quickly establish some ad-hoc system enabling existing devices to cooperate with one another, exchange data and/or code in order to coordinate activities of the emergency response personnel.

It is our goal to investigate the possibility of adapting our wireless infrastructure to enable collaboration among the emergency response and disaster relief personnel. For example, help to earthquake victims in a remote area without adequate network infrastructure can be administered promptly and quickly using handheld computing devices employed by rescue personnel.

In this paper we presented a framework for collaborative problem solving for mobile devices. It can be used for sharing of processing resources across a number of mobile devices to redistribute the computational load of a large and/or complex computing problem. The currently implemented simulation of the framework is aimed at studying the effects of different network configurations taking into account the device mobility, task

initiation frequency and the number of available mobile devices. We hope that the simulation results will demonstrate the advantages of the proposed collaborative problem-solving framework for computationally-intensive tasks.

7. REFERENCES

- [1] Bhagyavati, S. Kurkovsky. Wireless Grid Enables Ubiquitous Computing. In *Proceedings of The 16th International Conference on Parallel and Distributed Computing Systems (PDCS-2003)*, Reno, NV, Aug. 2003.
- [2] Foster et al. The Physiology of the Grid. In F. Berman, G.C. Fox, A.J.G. Hey (Eds.), *Grid Computing*, Wiley, 2003.
- [3] M. Gaynor, L.W. McKnight, J. Hwang, J. Freedman. Wireless Grid Networks and Virtual Markets. In *Proceedings of International Conference on Computer, Communication and Control Technologies (CCCT '03)*, Orlando, FL, July 2003.
- [4] H. Kuang, L. Bic and M. Dillencourt. Iterative Grid-Based Computing Using Mobile Agents. In *Proceedings of the 2002 International Conference on Parallel Processing*, Vancouver, B.C., Canada, Aug. 2002.
- [5] S. Kurkovsky, Bhagyavati. Agent-Based Distributed IDA* Search Algorithm for a Grid of Mobile Devices. In *Proceedings of The 7th World Multi-Conference on Systemics, Cybernetics and Informatics (SCI-2003)*, Orlando, FL, July 2003.
- [6] S. Kurkovsky, Bhagyavati. Modeling a Computational Grid of Mobile Devices as a Multi-Agent System. In *Proceedings of The 2003 International Conference on Artificial Intelligence (IC-AI'03)*, Las Vegas, NV, June 2003.
- [7] Z. Li, C. Wang, R. Xu. Computation Offloading to Save Energy on Handheld Devices: A Partition Scheme. In *Proceedings of International Conference on Compilers, Architectures and Synthesis for Embedded Systems (CASES'01)*, Atlanta, GA, Nov. 2001.
- [8] L.W. McKnight, J. Howison. Towards a Sharing Protocol for Wireless Grids. In *Proceedings of International Conference on Computer, Communication and Control Technologies (CCCT '03)*, Orlando, FL, July 2003.
- [9] T. Phan, L. Huang, C. Dulan. Challenge: Integrating Mobile Wireless Devices into the Computational Grid. In *Proceedings of The 8th International Conference on Mobile Computing and Networking (MOBICOM'02)*, Atlanta, GA, Sep. 2002.
- [10] A. Ray, Bhagyavati. Mobile Devices in a Computational Grid. In *Proceedings of The 41st ACM Southeast Regional Conference (ACMSE-03)*, Savannah, GA, March 2003.
- [11] J.P.G. Sterbenz et al. Survivable Mobile Wireless Networks: Issues, Challenges, and Research Directions. In *Proceedings of the ACM Workshop on Wireless Security*, Atlanta, GA, September 2002.
- [12] O. Tomarchio, L. Vita. On the Use of Mobile Code Technology for Monitoring Grid System. In *Proceedings of First IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGrid 2001)*. Brisbane, Australia, May 2001.