Design and Implementation of a Personal Super Computer

Heba A. Kurdi

Al Imam Muhammad Ibn Saud Islamic University, Riyadh, SA

ABSTRACT

Resources of personal devices, whether mobile or stationary, can be productively leveraged to service their users. By doing so, personal users will be able to ubiquitously run relatively complex computational jobs, which cannot be accommodated in their individual personal devices or while they are on the move. To this end, the paper proposes a Personal Super Computer (PSC) that superimpose grid functionality over networked personal devices. In this paper, architectural designs of (PSC) were developed and evaluated thoroughly through a strictly controlled empirical evaluation framework. The results showed that this system has successfully maintained high speedup over regular personal computers under different running conditions.

KEYWORDS

Grid computing, personal networks, mobile devices, resource scheduling.

1. INTRODUCTION

The Advances in fabrication density and microprocessor design have shrunk the sizes and weights of devices. This has enabled individuals to move with a number of electronic gadgets (cellular phone, GPS navigator, PDA, laptop etc.). With the available wired technologies, such as USB and Firewire, as well as wireless technologies, for example the IEEE 802.15 family of standards, these devices can be connected to form a Personal Area Network (PAN) with the user at its inner core [1].

Besides this basic set of electronic devices within the PAN, one might have other devices in different locations, for instance in the home, office and car. These devices, which belong to the same user, can be connected together regardless of their geographical locations to form a Personal Network (PN) [2]. Thus, one can gain access to his/her electronic devices, any time anywhere, and can share resources among them.

Since PNs that allow personal users to share data and peripherals are already there, the next logical step is to allow sharing of all other network resources including processors cycles and runtime memories, in other words, to superimpose the functionality of computational grids on top of PNs to form a Personal Super Computer, as shown in Figure 1.

In Grid computing computational resources are combined to form a large-scale distributed system in which all resources, including processor cycles and memories, might be shared [3]. However, notwithstanding the escalating popularity of grid computing in both research and enterprise domains, personal users, i.e. individuals outside these domains, are still not supported [4].

DOI:10.5121/ijcsit.2014.6209

Furthermore, ideally, to deploy a grid, powerful computational resources are combined and shared. As a consequence, many researchers consider mobile devices as at best only marginally relevant to grid computing. This is due to:

- Typical limitations of these devices, in terms of: processing capability, persistent storage, runtime heap, battery lifetime, input methods and screen size, relative to stationary devices.
- High security risks and critical privacy requirements.
- Great heterogeneity and non-interoperability in terms of hardware, OSs and applications.
- Unreliable intermittent connectivity with low bandwidth.
- Highly demanding applications as applications intended to be executed in mobile devices should be designed carefully such that their problem space is decomposable and distributable among several devices.

However, [3] necessitated the scaling of grids to both a large number of entities and to smaller devices. There are many indicators supporting this necessity. First, every measure of the capabilities of these devices including processing speed and memory capacity, is improving, and expected to continue, at exponential rate following Moore's law of increasing transistor density [5]. Second, the number of mobile devices in the world is escalating and expected to soon dominate the number of personal computers [6]. Indeed, the Wireless World Research Forum (WWRF) predicts that there will be 1000 wireless devices per person on average in 2017 [7]. Third, in many emergency situations, such as natural disasters and fire fighting, mobile devices might be the only accessible communication and computation tools. Fourth, the wireless connectivity and availability is improving as seen in current 3G networks.

Therefore, this paper aims at creating a means to bridge the gap between computational grids and personal users with resource limited mobile devices in a form of a personal mobile grid system which we called is Personal Super Computer (PSC).

The remainder of the paper is organised as follows. Section II proposes architectural designs for Personal Su. Section III explains the research methodology followed during this research. Section IV presents and analyses the simulation results then Section V concludes the paper.

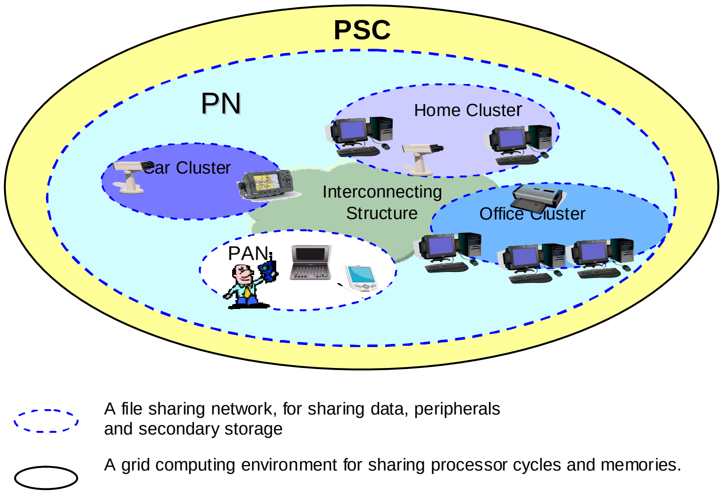

PSC
PN
Home Cluster
Car Cluster
Interconnecting Structure
Office Cluster
PAN
A file sharing network, for sharing data, peripherals and secondary storage
A grid computing environment for sharing processor cycles and memories.

Figure 1: Personal Super Computer

2. PERSONAL SUPER COMPUTER (PSC)

2.1. What is a PSC

A PSC is a personal mobile grid environment which can be owned and utilised by an individual user. It is constructed over his/her devices and might be extended to other devices which s/he trusts. PSC aim to enable the mobility of both, users requesting access to grid resources and resources that are part of a grid. Hence, the distinguishing characteristic of a PSC is that it is primarily constructed, owned and utilised by an individual (or a group of individuals with a mutual trust relationship). This is in contrast to traditional grids which are constructed, owned and utilised by organisations and other large entities. In other words, where traditional grids are concerned with a large user population, the PSC is only concerned with a single user. Also the type of application is different; where traditional grids are chiefly concerned with massive complex world-wide computations, PSC applications are considerably smaller in size, scope and complexity. Additionally, where traditional grids need a well-established stationary infrastructure to operate, PSC can be fully accommodated in mobile devices connected via a PN.

2.2. System Design

PSC are designed based on PN architecture and as a natural extension to them. A PSC can be viewed as a superset of PNs. It is a PN with additional resources for sharing: CPU cycles and run-time memories, which allow for additional public and private services.

Abstract layered architecture

The PSC architecture is based on the three layers (levels) PN architecture [2]. An additional layer is introduced between the network and service layers, namely the PSC layer. Hence, the PSC architecture is composed of four abstract layers: the connectivity layer, network layer, PSC layer and the service layer. These layers act as a middleware system offering an abstraction over physical devices.

The PSC layer serves as a virtualisation layer to hide the complexity of harnessing the heterogeneous underlying computational resources from the end user. In this layer, resources available from the network layer are grouped into two main categories: personal resources and foreign resources based on the type of a trust relationship between these resources.

Personal resources are grouped into larger virtual resources based on the type of functionality they provide such as CPU cycles, storage, address book and printing.

Detailed Architecture

A PSC consists of groups of devices which are usually owned and utilised by the same person. All these devices are connected via a well secured network PN. Issues related to connectivity are tackled in the PN connectivity layer. Issues related to security and clustering are all handled at the PN network layer, while issues related to presentation and quality of services are dealt with at the PN service layer.

Thus, basically, the key missing functional component after superimposing grid functionality on top of a PN is a resource management system for the newly added grid resources represented by CPU cycles and runtime memories, as these resources require special handling to jointly execute computational jobs in PSCs. The main functions of this resource management system is to decompose parallel jobs, if possible, into smaller tasks that can be accommodated by mobile devices, then mapping these tasks to proper resources and, after execution, composing final results sending them back to clients. Therefore, as shown in Figure 2, a PSC includes three functional elements: clients, workers and spaces.

Typically, clients are mobile devices, such as mobile phones, usually within the PAN, that are highly dynamic and considerably limited in terms of processing power and network bandwidth. This set can send requests for executing simple jobs or complex computational jobs that are stored elsewhere, to more capable devices in PSC.

Workers represent devices that can be mobile but are less dynamic and have better computing resources than clients, such as laptops. These devices can jointly complete computational jobs. They are divided further into:

- Executers: These are computing elements capable of executing the actual computation logic encapsulated in a job after decomposing it into smaller fine grain tasks.
- Composers: Since jobs are decomposed into smaller tasks, and each task is executed independently of other tasks within the same job, there is a need to aggregate results produced after running these tasks. Composers are elements running a specialised program that compose all partial results related to a certain job into a final result making it ready to clients.

Spaces consist of a set of static storage-rich devices mainly at home or the office, such as desktops. Clients and workers communicate with each other using these spaces which serve basically as simple shared memories for buffering. The use of a buffering technique is important in mobile environments to reduce the impact of frequent disconnections. The idea of spaces is based on Tuple-spaces first realised in the Linda system language. A Tuple-space is a form of independent associative memory. For example, consider a group of processors that produce pieces of data and a group of processors that consume the data. Producers post their data to the space, consumers retrieve data from the space that matches certain criteria. In PSCs, there are two types of spaces:

- Work-spaces: Work-spaces are multiple pools of jobs sent from clients. Executers access these pools, hunting for tasks to execute.
- The result-space: the result-space is a large pool holding results that are generated by executers.

In Figure 2 the hierarchal relationship between the main elements of a PSC is illustrated. Although, each element had been defined earlier as a set of devices, an element actually represents a logical role which is a functionality that can be added to any device in a PSC, based on its capabilities. Roles are “upward compatible” where workers can act as clients while spaces can act as workers and clients as well. During initialisation, each device is assigned an initial role based on its score in the Device Score (DS) formula:

$$DS = w_1A_1+w_2A_2+w_3A_3+\ldots+w_nA_n \qquad (1)$$

where A_1, A_2, $A_3,\ldots,A_n$ are the set of static normalised attributes relevant to the device performance, such as CPU speed, memory size, network bandwidth, immobility and remaining power in the battery . The weighting coefficients w_1, w_2, $w_3,\ldots,w_n$ are used to describe the relative importance of the different device attributes in each role, subject to:

$$\sum_{i=1}^{n} w_i =1,(2\ ) \text{ and } w_1, w_2,\ldots,w_n \geq 0 \quad (3)$$

At the operation time, a device might be promoted (assigned a higher role in the PSC roles hierarchy) based on the device score in the DS formula after substituting A_1, A_2, A_3,…,A_n by the device dynamic attributes such as current CPU load, available memory and battery consumption. For instance a laptop with a low battery might be promoted to a client.

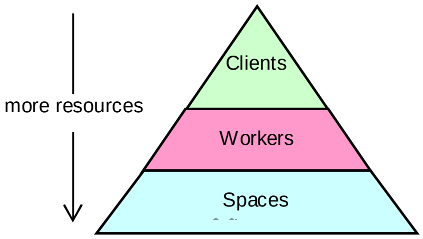


Figure 2: PSC Components

3. Evaluation Methodology

The research started by surveying the area of grid computing and distributed systems for paradigms relevant to PSCs. The survey revealed two main findings. First, there are few research projects which have addressed the mobility issue in grid computing [8] but only at the organisational level. Second, fewer research projects have targeted grid systems at the personal level [9, 10], but the focus has only been on facilitating file sharing applications. Therefore, an architectural design of PSCs was developed to address both personalisation and mobility issues in grid computing.

The most important aspect of realising a grid system is a scheduler that efficiently utilises its resources. However, the extremely dynamic nature, diversity and limited capabilities of resources, as well as difficulties in predicting the nature and timing of incoming jobs, are all factors that increase the complexity of the scheduling problem in PSCs.

Therefore, a survey on resource scheduling frameworks was conducted to address design features required for a resource scheduler that can cope with the extraordinarily difficult scheduling conditions in PSCs. The survey revealed that decentralised, cooperative, local, adoptive, non-clairvoyant and self-scheduling schemes are among the top requirements to deal with the complexity of this problem. Consequently, a resource scheduler, HoPe: Honeybee inspired resource scheduler for Personal mobile grids, was proposed and implemented based on these requirements. HoPe was augmented with techniques analogous to those utilised by the honeybee colony, while allocating worker bees to nectar sources under the extraordinarily difficult conditions of weather unpredictability and food variability.

Next, PSC designs and HoPe implementation were evaluated thoroughly through a strictly controlled empirical study considering two main grid design issues: scalability to a larger number of nodes and sustainability under different loads.

The controlled study involved the following steps:

1. Identifying the critical elements inherent in the design of grid systems and deciding on the set to be considered: job interarrival time, number of nodes, job size and processor capacity.
2. Varying the experimental variables, job interarrival time and number of nodes, to simulate a representative sample of grid environments. Values of the number of nodes (workers) were selected in the range of the expected number of devices per cluster available for an

individual user or a small business: from 4 to 16 nodes. Values of interarrival time were selected in the range of two extreme cases of the expected usage of PSCs: from 4 to 180 sec.

3. Controlling extraneous variables, job size and processor capacity, by randomisation to ensure a representative sample in all experiments. Heterogeneity in processor capacity was modelled assuming three types of machines (P_a, P_b, $P_{c)}$ with different capacities. Heterogeneity in job size was modelled assuming three types of jobs (J_a, J_b, $J_{c)}$ with different sizes. During running time, a uniform random number R_{proc} from one to three was generated describing the processor capacity and another random number R_{job} following the same distribution was generated to describe job size heterogeneity. Basically, the processor capacity and job size were generated based on similar lines of research conducted by [7].
4. Identifying a benchmark algorithm. The opportunistic scheduling heuristic (OSH), a well established heuristic in high throughput computing, was selected for this purpose.
5. Identifying suitable performance measures: Speedup which refers to how much a parallel system is faster than a corresponding sequential system was used as the main performance measure

$$S_p = T_1 / T_p \qquad (4)$$

$$T_p > 0$$

where:

p is the number of processors

T_1 is the execution time of the sequential algorithm

T_p is the execution time of the parallel algorithm with p processors.

Although speedup is usually calculated based on one job, in the case of HoPe and OSH, calculating the speedup in this way would be out of context, as these heuristics operate in a steam of jobs. Therefore, the mean time of speedup is considered.

6. Comparing the performance of both HoPe and OSH to optimum values and lower bounds and analysing the main findings.
7. Improving the accuracy of this simulation-base study through: running ten simulations and accepting the mean outcome; ignoring simulation results generated in the first 60 sec; measuring uncertainty in data using the standard deviation; and calculating absolute and relative errors to examine the quality of results.

4. Results and Discussion

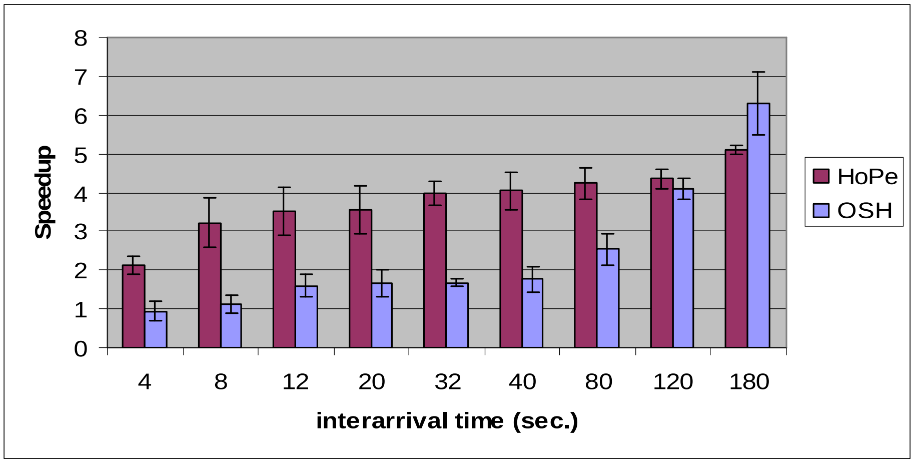
Simulation results obtained after running each experiment for five hours. Jobs were generated using four clients with a Poisson process and exponential interarrival times with means in the range from 4 to 180 sec. Computational jobs were implemented as divisible load cryptography applications to factor large integers (up to 4,293,001,441). Each job was contained in one packet and produced one output file. For simplicity, the communication cost was not considered at this stage. It was assumed that one machine can process only one operation at a given moment (resource constraints) and once task started, operation runs to completion (no pre-emption condition).

Figure 3 consists of three sub-figures showing the speedup of both HoPe and OSH at the three grid scales. The speedup is calculated based on equation (4) with the empirical value of 27 sec. as the execution time of the sequential algorithm. A worst bound of one is assumed, representing the case when both running times of executing a job sequentially, in one machine, and in parallel machines, are equal.

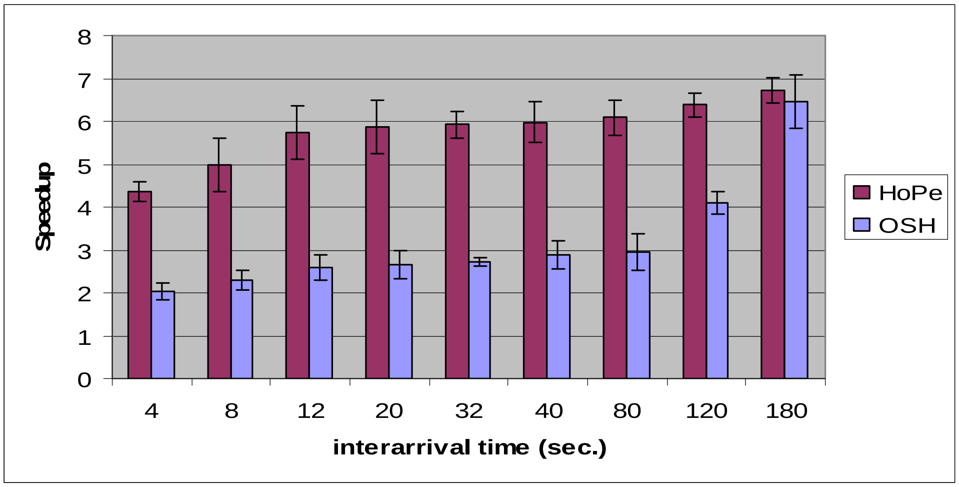
The figure shows that HoPe has maintained a noticeably higher speedup which reaches the double speedup of the OSH in nearly 60% of all scenarios. However, the difference between the two heuristics in performance decreases gradually as the interarrival time gets larger in small and medium grid scales.

HoPe has its speedup values in the range from 2 to 10 which is double the speed of the sequential execution (worst bound) in its worst case and ten times faster than the sequential execution at best. The speedup of the OSH lies in the range from 0.9 to 7 which is a slowdown in its worst case and, in its best case, it is only seven times faster than the sequential execution.

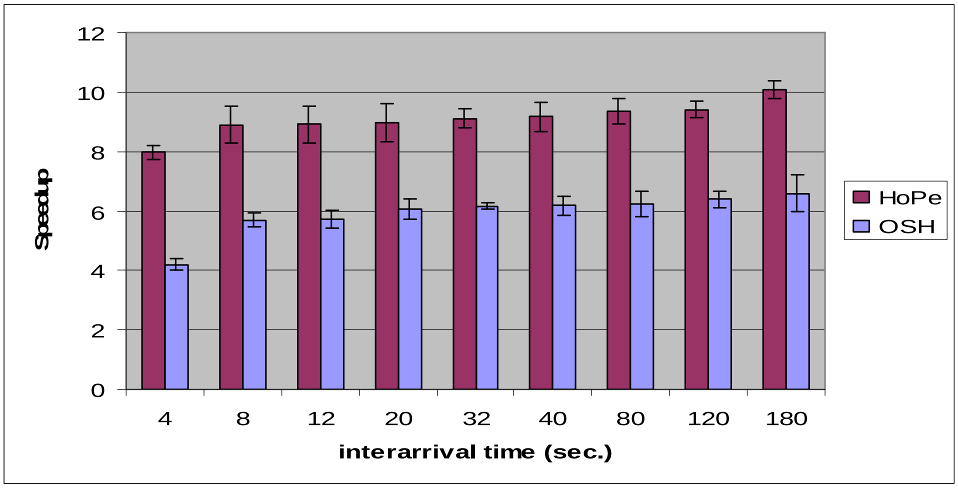
As expected, the speedup of both HoPe and the OSH is highly affected by the grid scale in terms of the total number of worker devices in the system. The interarrival time has a lower effect when HoPe is considered.



(a) 4 workers per cluster



(b) 8 workers per cluster



(c)16 workers per cluster

Figure 3: Average Speedup in HoPe and OSH

5. CONCLUSIONS

In this paper, Personal Super Computers (PSCs) were introduced as Personal Mobile Grid environments owned and utilised by individual personal users. An architectural design of PSC based on PNs architectures was proposed. and evaluated. The results indicate the potential of PSCs in realising futuristic grid visions to scale the grid entities to larger number of entities and smaller devices and to widen the grid application areas to span more geographical and social settings than ever before. Therefore considering the deployment of PSCs in real life scenarios is highly recommended.

REFERENCES

[1] IEEE 802.15 Working Group for WPAN [online]. Available: http://ieee802.org/15/, [accessed Dec 12, 2013].

[2] My Personal Adaptive Global NET (MAGNET) (IST 507102) [online]. Available: http://www.ist-magnet.org, [accessed Dec. 14, 2013].

[3] I. Foster and C. Kesselman, Eds., The Grid2: Blueprint for a Future Computing Infrastructure. San Francisco: Morgan Kaufmann, 2003.

[4] J. Han and D. Park, “A lightweight personal grid using a supernode network”, in Proc. 3rd Int. Conf. P2P 2003, pp. 168-175.

[5] K. Michael, (2003, Feb. 10) Moore's law to roll on for another decade, cnet news [online]. Available: http://news.cnet.com/2100-1001-984051.html, [accessed Dec. 12, 2013].

[6] Ipsos Insight, “Mobile phones could soon rival the PC As world’s dominant Internet platform,” Market study rep., summary, Apr. 2006 [online]. Available:http://www.ipsos-na.com/news/pressrelease.cfm?id=3049, [accessed Dec. 12, 2013].

[7] N. Jefferies, “Global vision for a wireless world,” 18th meeting of WWRF, 2007, Helsinki, Finland.

[8] Akogrimo [online]. Available: http://www.mobilegrids.org/, [accessed Feb. 2, 2010]..

[9] J. Han and D. Park, “A lightweight personal grid using a supernode network,” in Proc. 3rd Int. Conf. P2P2003, pp. 168-175.

[10] W. Li, Z. Xu, B. Li, Y. Gong, “The Vega Personal Grid: A lightweight grid architecture,” in Proc. IASTED, 2002, pp. 6-11.

Authors

Dr. Heba Kurdi received her B.Sc. and MSc. degrees in computer science from King Saud University Riyadh, Saudi Arabia, in 1999, and 2003 respectively. In 2010, she got her PhD in wireless networks and communications from the School of Engineering and design, Brunel University, UK, where she also contributed as a visiting researcher in Summer 2011. She is currently an assistant professor at Al Imam Muhammad Ibn Saud Islamic University, SA. She has many publications in international conferences and journals and she is also a frequent reviewer for several computer science journals. Her research interest includes grid and cloud computing, and Bio-inspired Engineering.