Journal of Ubiquitous Systems & Pervasive Networks
Volume 6, No. 1 (2015) pp. 01-10

Personal Mobile Grids: Ubiquitous Grid Environments For Personal Users

Heba Kurdi*

King Saud University, Riyadh , SA, 11432

Abstract

The overall aim of this paper is to introduce Personal Mobile Grids (PM-Grids) as a novel paradigm in grid computing that eases scaling grid infrastructures to mobile devices and extending grid users to individuals outside research and enterprise domains. In this paper, architectural designs and simulation models for PM-Grids are presented as well as a honeybee inspired resource scheduling heuristic incorporating a radical approach to grid schedulers. A detailed design and implementation of HoPe with a decentralised self-management and adaptive policy are presented. PM-Grid designs and HoPe implementation were evaluated thoroughly through a strictly controlled empirical evaluation framework with a well-established heuristic in high throughput computing, the opportunistic scheduling heuristic (OSH), as a benchmark algorithm. Experimental results demonstrated the superiority of HoPe performance in terms of stability, throughput and turnaround time, under different running conditions of number of jobs and grid scales.

Keywords: *Distributed networks, Mobile Computing Architectures, Mobile environments, Ubiquitous computing*

1. Introduction

Nowadays, the mobile devices market has become one of the largest markets in the world. It is rapidly evolving with progressive reduction in cost, weight and size and continuous improvement in performance. This has enabled many people to move around with a basic set of electronic gadgets such as mobile phones and laptops. These devices, which belong to the same user and are usually within ten metres of her/him, can be connected together with the user at its inner core forming a Personal Area Network (PAN) [1],[2]. 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) [3][4]. Thus, one can gain access to his/her electronic devices, any time anywhere.

Nonetheless, PANs and PNs are most commonly used for applications involving data and peripheral sharing. This is due to the resources allowed for sharing in PNs, PANs and all today's conventional networks being limited to data, peripherals and secondary storage. The most important resources, namely, processors cycles and runtime memories, are still not available for sharing across these networks.

Hence, an important question arises here: Why not further enable these networks to seamlessly share all resources and functionalities in the form of services available across computational grids [5]. As PNs can already share data, peripherals and secondary storage among their devices, the next logical step is to superimpose grid functionality over them to allow the sharing of processors cycles and memories. Thus, the net result is a huge virtual computer, which can be accessed at anytime from anywhere. That is to say, a Personal Mobile Grid (PM-Grid).

The core of any grid systems is an efficient scheduler that offers better management and utilization of virtualized underlying infrastructure resources [6]. Therefore, this paper aims at designing, implementing and evaluating a PM-Grid with a honeybee inspired resource scheduler (HoPe). The proposed scheduler is based on a non-clairvoyant scheduling policy, hence being able to deal with realistic demands, where incoming requests for computing resources are unpredictable in terms of timing and nature and running environments are dynamically changing. This is considered as a great improvement over current grid schedulers as they are of clairvoyant policies assuming availability or predictability of information about incoming jobs and running environments. Clairvoyant policies drastically limit schedulers flexibility in managing cloud infrastructure and considerably increase their running time overheads in order to collect information about jobs and underlying infrastructures or make reasonable prediction about them.

PM-Grid designs and HoPe implementation were evaluated thoroughly through a strictly controlled empirical evaluation framework with a well-established heuristic in high throughput computing, the opportunistic scheduling heuristic (OSH), as a

* Corresponding author. Tel.: +966555669791
Fax: +9876543210; E-mail: hakurdi@ima,u.edu.sa

DOI: 10.5383/JUSPN.06.01.001

benchmark algorithm. Comparisons with optimal values and worst bounds were conducted to gain a clear insight into HoPe behaviour, in terms of stability, throughput, turnaround time and speedup, under different running conditions of number of jobs and grid scales. Experimental results indicate the efficiency of PM-Grid designs and demonstrate the ability of HoPe to considerably reduce the effect of variations in grid scale and job inter arrival times, illustrating better scalability and sustainability, when compared to the OSH.

The rest of this paper is organised as follows: section 2 describes the design of PM-Grids at abstract and detailed levels. Section 3 introduces the HoPe resource scheduler as the core element for PM-Grids. In section 4, the evaluation methodology and the experimental process are detailed while results and discussion are presented in section 5. Section 6 compares PM-Grids with related projects and finally, the paper is concluded in section 7.

2. PM-Grid Architecture

A PM-Grid is a grid environment, which can be owned and utilised by an individual user. It is constructed over her/his devices and might be extended to other devices which s/he trusts. PM-Grids aim to enable the mobility of both, users requesting access to grid resources and resources that are part of a grid. This opens the doors to have the grid processing power in more geographical locations such as emergency communications in firefighting and natural disasters, as well as many of the newly emerged mobile applications in e-learning, e-healthcare among others.

2.1 Abstract Layered Architecture

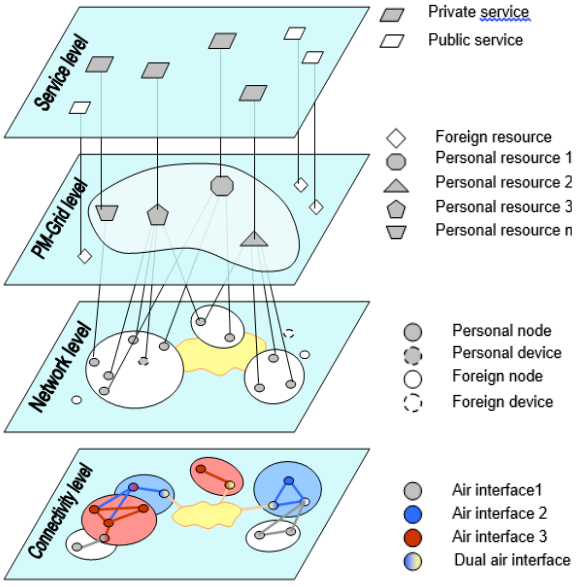

Fig. 1. PM-Grid layered Architecture based on the PNs three level architecture with the additional PM-Grid Level.

The PM-Grid architecture is based on the three levels PN architecture proposed by the MAGNET project [4]. An additional level is introduced between the network and service levels, namely the PM-Grid level, as shown in figure1. Hence, The PM-Grid architecture is composed of four abstract levels: the connectivity level, network level, PM-Grid level and the service level. These levels act as a middleware offering an abstraction over physical devices.

The added PM-Grid level serves as a virtualisation layer to hide the complexity of harnessing the heterogeneous underlying computational resources from the end user. In this level, resources available from the network level are grouped into two main categories: personal resources residing inside the PM-Grid, and foreign resources residing outside the grid.

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. The aim is to allow personal users to submit service requests, for example, a request for CPU cycles and memory to execute a computational job, from any device available within their trusted PNs without being concerned about where/when/how these requests are executed. To achieve this goal, the grid level should provide an efficient resource scheduler. The scheduler is responsible for automatically decomposing, allocating and executing jobs, then finally composing final results, making them ready for the end user. The scheduler should be lightweight, self-managed and adaptive to cope with the dynamic nature of the PM-Grid environment. A detailed design of such a resource scheduler is presented in section 3.

2.2 Detailed Architecture

A PM-Grid 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 level. Issues related to security and clustering are all handled at the PN network level, while issues related to presentation and quality of services are dealt with at the PN service level.

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. These resources require special handling to jointly execute computational jobs in PM-Grids. 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, from an architectural point of view, a PM-Grid consists of three functional elements: clients, workers and spaces.

Clients: devices with the least built-in resources, e.g. mobile phones. They are used mainly for sending jobs or/and receiving results.

Spaces: devices with high storage capabilities, e.g. media players. They serve as a simple associative memory where entities communicate with each other. Spaces are further divided into:

- Work-spaces to hold submitted jobs.
- A result-space that contains intermediate and final results.

Workers: devices with high processing capabilities, e.g. laptops and desktops. Workers are subdivided into:

- Executors to run the computational logic encapsulated in a job.
- Composers to collect all initial results related to a certain job and compose them into a final result.

The special organisation for distributing the system functionality among multiple agents (workers) with a single target pool (result-space) and multiple job sources (work-

spaces) are inspired by the way honeybees are organised in a colony, as explained in section 3.

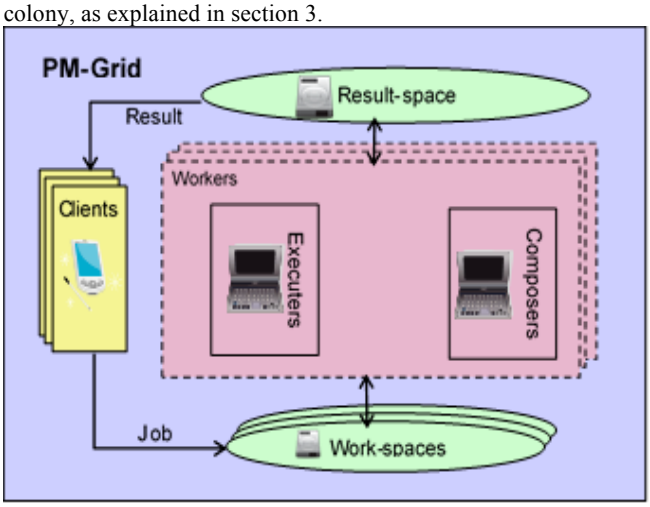

Fig. 2. PM-Grid Detailed View consists of three functional elements: clients, workers and spaces.

3. Resource Scheduler Design

3.1 Scheduling Problem in PM-Grids

A PM-Grid is a unified collection of resources connected via a PN. It has the potential to deliver grid level services to a personal end user. Whenever a need occurs, a PM-Grid user uses his/her client device to send a computational job for execution on his/her PM-Grid. The job is received at the nearest work-space. Hence, a variable unpredictable stream of incoming jobs arrives at each work-space from client devices. Executer devices need to be efficiently allocated to incoming job streams producing results that are sent to the result-space where an unpredictable stream of generated results arrives. All results that belong to one job are accumulated in a separate output file. When an output file containing all job results is ready, it is allocated to a composer device for final preparation before being dispatched to the sender or a requested address.

As in the case of all grid systems, the core of a PM-Grid is a scheduler, which strives to efficiently assign tasks to available grid resources. Grid resource scheduling is a complex problem in general [7]. Centralised plan-ahead schedulers are usually deployed for this purpose [8]. In these schedulers, a single authority is in charge of all decisions regarding who should run what and when. Two assumptions are common in such schedulers: First, clear and sufficient information about incoming jobs is known in advance, which is simply not realistic. Second, a globally detailed and frequently updated view of the system resources state is available [9], which is prohibitively expensive, severely restricts the scalability of the system and exposes it to high security risks.

Assuming the availability of clear information about the incoming jobs before making the scheduling decision, is what is referred to as clairvoyant scheduling, with which virtually all grid resources are concerned. Although this clairvoyant assumption considerably simplifies the scheduling problem, it is not valid for most real world problems [10]. In contrast, the non-clairvoyant scheduling approach assumes that such information is unavailable in advance, making it more practical for many computer engineering problems, especially grid computing where it is usually difficult and costly to make reasonable predictions.

The scheduling problem in a PM-Grid can be defined as efficient non-clairvoyant scheduling in a highly dynamic environment of limited resources. The non-clairvoyant scheduling problem is considered as NP-hard as it contains two classical NP-hard problems as special cases:

- The first case, when all tasks are sequential, the problem reduces to the multiprocessor scheduling problem, which is NP-hard [11].
- The second case, when all tasks have the same execution time, the scheduling problem becomes the bin-packing problem, which is NP-hard also [12].

Therefore, one practical way to solve this problem is to design a heuristic that tries to find a "good" solution for this extraordinarily difficult scheduling problem [13].

3.2 Scheduling Problem in Honeybee Colony

A honeybee colony has a limited number of bees, which it needs to allocate wisely to the surrounding flower patches from which they collect nectar and bring it to the hive for further processing in order to generate comb honey. This process is what has been referred to as the Nectar Acquisition Process (NAP).

During NAP, a honeybee colony divides labour, based on temporary specialisation, between two groups: forager bees, who work in the field collecting nectar from food sources turning it into raw honey, and receiver bees, who work in the hive processing raw honey to produce comb honey (honey-filled wax comb as stored directly by the bees). This organisation boosts the efficiency of the NAP, but requires dynamic coordination of the two labour groups to keep the rates of nectar collection and honey processing in balance.

This coordination problem is significant because the colony experiences large and unpredictable variations in the nectar availability. The colony adjusts its nectar collection and honey processing rates with respect to external nectar supply mainly by dynamically adjusting the number of forager and receiver bees through "waggle" and "tremble" dances.

When food sources are laden with nectar, the colony increases the number of forager bees, raising the nectar collection rate. This is done through the waggle dance, which stimulates some receiver bees to change their roles to foragers and help in nectar foraging. On the other hand, when the processing rate is lowered, having a number of receiver bees changed their role to forager bees, the colony speeds up the honey processing rate through tremble dance. The tremble dance stimulates some forager bees to work as receiver bees [14].

So basically, the honeybee colony faces an extraordinarily difficult scheduling problem in nature, due to weather unpredictability and food variability, while allocating honeybees to nectar sources during the NAP. The colony efficiently solves this problem through simple non-intelligent agents, (honeybees) running a decentralised cooperative and adaptive self-scheduling policy. The aim is to maximise the nectar intake while maintaining the hive at a stable state where nectar collecting and honey processing rates are balanced. This observation is the foundation of the broad hypothesis behind The Honeybee Inspired Resources scheduler for Personal Mobile Grids (HoPe): Efficient non-clairvoyant scheduling in a highly dynamic environment of limited resources may be achieved with a heuristic approach based on simple agents. The agents allocate themselves to multiple work sources in a decentralised, cooperative and adaptive self-scheduling scheme striving to maximise work intake while maintaining the system in a stable state, in an attempt to imitate the behaviour of honeybees during the NAP. The mapping between the main

honeybee colony and HoPe elements is presented in Table 1 and HoPe algorithm is detailed in [15][16].

TABLE 1. The Honeybee Analogy

Honeybee colony	HoPe
Food sources	Work-spaces
Hive	Result-space
Forager bees	Executer devices
Receiver bees	Composer devices
Nectar	Job
Honey	Result

4. Evaluation

4.1 Evaluation Objectives

The thesis is that a PM-Grid can allow personal users to seamlessly combine their own personal devices, either mobile or stationary, to accomplish relatively large computational jobs. To test this thesis, an adaptive self-scheduling heuristic, HoPe, has been developed with an end aim of evaluating PM-Grids as a proof-of-concept. The aim has been fulfilled through the following objectives:

- Test HoPe performance by exploring how it is affected by variations in PM-Grid environment specifications, namely:
  - The job inter arrival time: The system should sustain various loads as personal users' requirements vary significantly.
  - The number of nodes: the system should be sufficiently scalable to accommodate different infrastructure scales, as PM-Grids can be utilised by individuals as well as small size organisations.
- Evaluate HoPe efficiency by comparing it to a well-established heuristic in the same area, as well as an optimum value or worst bound, when possible, for each performance metric.
- Build performance models for both heuristics to obtain a clearer insight into HoPe behaviour.

4.2 Experimental Design

There are two main limitations in the simulation methodology of current scheduling research. First, there are no simulation standards and, second, traditional computing platform standards are no longer valid for modern platforms [17]. To overcome this problem, strictly controlled experiments in a logical network model of PM-Grids have been designed which involved the following steps:

1. Identifying the critical elements inherent in the design of grid scheduling systems and deciding on the set to be considered in this experiment: job inter arrival time, number of nodes, job size and processor capacity.
2. Varying the experimental variables, job inter arrival time and number of nodes, to simulate a representative sample of grid environments.
3. Controlling extraneous variables, job size and processor computational capacity, by randomisation to ensure a representative sample in all experiments.
4. Identifying a benchmark algorithm. The opportunistic scheduling heuristic (OSH) has been selected for this purpose.
5. Identifying suitable performance measures, stability, net throughput, mean and Turnaround (TT), to compare HoPe and OSH.
6. Building a flexible PM-Grid simulator that offers an easily controlled environment and robust experimental design.
7. Comparing the performance of both HoPe and OSH to optimum values or worst bounds, then reporting and analysing the main findings.
8. Improving the accuracy of the simulation-base study through:

- Running 10 simulations and accepting the mean outcome.
- Ignoring simulation results generated in the first 60 seconds.
- Measuring the uncertainty in data using the measure of standard deviation (SD) and displaying the values as error bars in all charts.
- Calculating the absolute error and relative error to examine the quality of obtained results.

4.3 Resource Model

A simulation model of the PM-Grid platform was developed using Opnet™ [18] modeller. Three representative infrastructure scales of PM-Grids in potential application areas were considered:

- Small (4 workers/cluster).
- Medium (8 workers/cluster).
- Large (16 workers/cluster).

The model was simulated as a logical network, that consisted of N=5 clusters, as shown in Figure 3. All clients were placed in one cluster (cluster 0) which represented the PAN with the user at its inner core submitting jobs to his/her PM-Grid via devices in this cluster. For simplicity, the result-space was placed alone in a separate cluster (cluster 4). All other clusters consisted of one work-space and w workers.

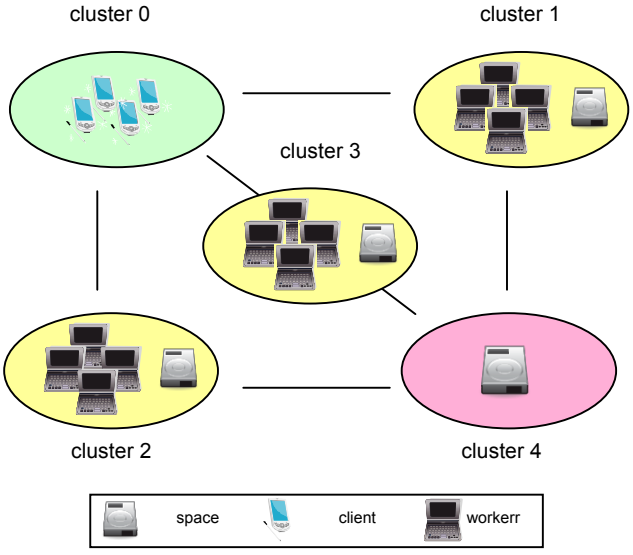

Fig. 3. Simulated Model for a small size PM-Grid environment with 4 workers/cluster.

From the total number of workers, 75% were initially assigned an executer role, and the remaining 25% were assigned a composer role. This selection was aimed to conform to the natural distribution of roles in a honeybee colony where [14] stated that nearly 75% of honeybees are food foragers. This model can scale easily and allows the testing of HoPe performance in isolation of possible effects caused by physical

hardware, network topology and implementation technologies. This isolation is important to gain a clear insight into HoPe performance. Experimenting with realistic networks is left for future work to see how physical hardware and network parameters of a PM-Grid may affect HoPe performance.

4.4 Job model

The job model assumed by HoPe is divisible workload (DL) applications where each job can be divided into an arbitrary number of independent tasks of low granularity. It is assumed that the input to each task is a single file, which is sent with the task. Each task produces exactly one output file, as shown in Figure 4. This model can be found in many everyday application areas related to personal users such as image processing, database searching and cryptography.

Without loss of generality, this paper has considered a cryptography application, based on the Trial Division Algorithm, in particular as it has potential applications in personal environments where security and privacy are critical issues. All worker devices are assumed to have a word-size of, at least, sixteen-bits. The last prime that fits into a sixteen-bit unsigned integer should be less than 2^{16}-1=65,535, which is 65,521. That suffices to factorise numbers up to $65{,}521^2$ = 4,293,001,441.

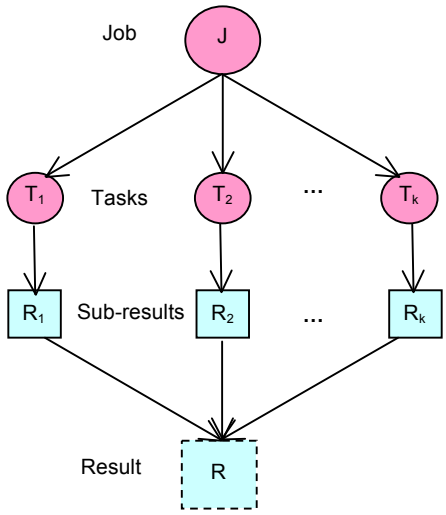

Fig. 4. Job model assumed by HoPe is divisible workload (DL) applications where each job can be divided into an arbitrary number of independent tasks of low granularity.

The workload model is simulated as streams of DL jobs arriving at each work-space according to a Poisson process. Multiple values for both job size and inter arrival time are considered to ensure a representative sample in all experiments, as necessitated by [19]. The job size is considered as an intrinsic variable and controlled by randomisation. Heterogeneity in job size is modelled assuming three sizes of jobs (J_a, J_b, J_c). During running time, a uniform random number R_{job} from 1 to 3 is generated indicating the job size, as shown in Table 2.

Job inter arrival time is considered as an experimental variable; nine different values for inter arrival time were selected in the range between two extreme cases of the expected usage of PM-Grids: (4, 8, 12, 16, 20, 32, 40, 80, 120 and 180) seconds.

TABLE 2. Experimental Job Sizes

Job	Job size (j) in Mflop
J_a	$2\times10^2 < j \leq 3\times10^2$
J_b	$1\times10^2 < j \leq 2\times10^2$
J_c	$j \leq 1\times10^2$

4.5 Performance Metrics

Three performance metrics were considered:

Stability: where the system strives to maximise the job collection rate subject to minimising the difference between job collection and result generation rates. The stability is calculated as the absolute value of the difference between the job collection rate $F(N_c)$ and the result generation rate $F(N_p)$ as follows:

$$Stability = (1-| F(N_p) - F(N_c) |)\times100$$

Mean turnaround time (TT): represents the elapsed time from when a client submits a job until the client receives the corresponding results, and is calculated as:

$$TT = result\ received\ time - job\ submission\ time$$

Net throughput: represents the amount of work completed by the system over a period of time. It is measured as the number of jobs completed from time zero to time t.

5. Results and Discussion

Each scenario, simulating five hours (18000 sec.) of real time, ran several times and means were calculated after discarding data from the initial 60 sec. Mathematical and graphical performance models that illustrate HoPe and OSH behaviours, under different running conditions, were generated using multiple regressions and full quadratic equations. The models show the general behaviour of both heuristics when inter arrival time falls in the range from 4 to 180 sec. and the grid scale is in the range from 4 to 16 workers/cluster. Results were analysed using the ANOVA test and statistical significance of the full quadratic models predicted was evaluated using Fisher's statistical test (F) and F-significant (F-signif.).

5.1 Stability

Figure 5 and Figure 6 illustrate HoPe and OSH stability respectively, in terms of the difference in rate between job collection and result generation cycles calculated using the mean time. The model in Figure 5 shows that HoPe tends to maintain optimum stability (100%-98%) in a considerable wide area of the entire problem space. As expected, when there are enough workers, no matter how often jobs arrive. HoPe can maintain the difference between job collection and result generation at a minimum level. The situation changes gradually as the grid scale shrinks when stability becomes more sensitive to the inter arrival time. The insignificance P-value of all coefficients, in Table 3, emphasises that HoPe has successfully marginalised the effects of variations in the grid scale and the job inter arrival time when stability is considered.

The model in Figure 6 shows that the OSH tends to maintain optimum stability in a relatively small area of the entire problem space. It is also clear from the model, and also from the significance *P-value* of (b_1 and b_3) coefficients in Table 4, that the OSH is more sensitive to variations in the inter arrival time under all grid scales in the displayed range.

Mathematical equations and statistical data of the HoPe stability model and the OSH stability model are presented in Table 3 and Table 4 respectively.

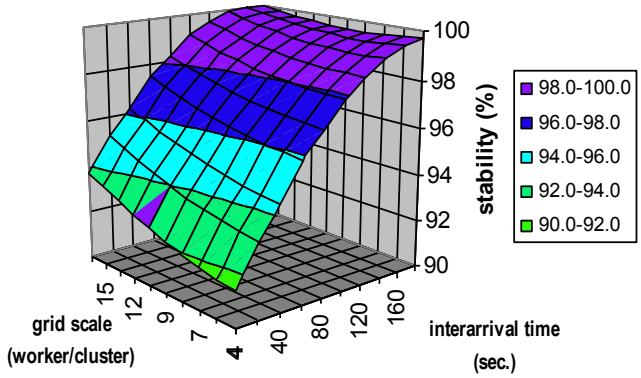

Fig. 5. HoPe stability model illustrating optimum stability (100%-98%) in a considerable wide area of the entire problem space.

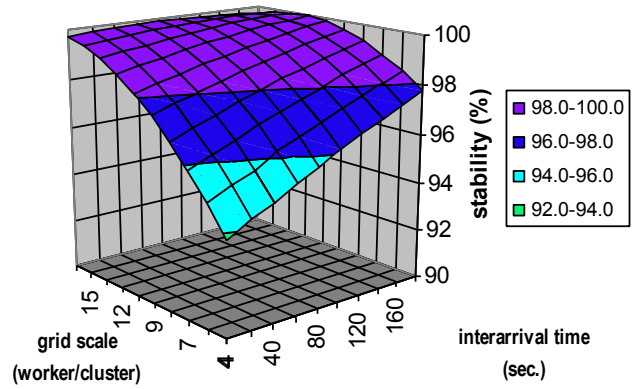

Fig. 6. OSH stability model illustrating optimum stability in a relatively small area of the problem space.

TABLE 3. Statistical Data of HoPe Stability Model

HoPe_stability $= b_0 + b_1 \times$ *inter arrival_time* $+ b_2 \times$ *grid_scale* $+ b_3$ *inter arrival_time* $\times$ *inter arrival_time* $+ b_4 \times$ *inter arrival_time* $\times$ *grid_scale* $+ b_5 \times$ *grid_scale* $\times$ *grid_scale*				
F =1.018 *F-signif* = 0.376				
Coefficients		***P-value***	**-95%**	**95%**
b_0	89.80	3.04E-24	86.16	93.43
b_1	0.03430	0.155	-0.00885	0.07744
b_2	1.085	0.268	0.262	1.909
b_3	-0.000022	0.339	-0.000236	0.000194
b_4	-0.001900	0.378	-0.00392	0.00016
b_5	-0.02900	0.385	-0.06805	0.01009

TABLE 4. Statistical Data of OSH Stability Model

OSH_stability $= b_0 + b_1 \times$ *inter arrival_time* $+ b_2 \times$ *grid_scale* $+ b_3$ *inter arrival_time* $\times$ *inter arrival_time* $+ b_4 \times$ *inter arrival_time* $\times$ *grid_scale* $+ b_5 \times$ *grid_scale* $\times$ *grid_scale*				
F = 3.159 *F-signif* = 0.02789				
Coefficients		***P-value***	**-95%**	**95%**
b_0	90.80	7.94E-22	86.09	95.51
b_1	0.09511	0.00385	0.03921	0.151
b_2	0.03720	0.960	-1.025	1.100
b_3	-0.000239	0.01638	-0.000517	3.86058E-05
b_4	-0.000977	0.432	-0.00362	0.00167
b_5	0.00800	0.836	-0.04264	0.05864

5.2 Net Throughput

Figure 7 and Figure 8 summarise the behaviour of HoPe and the OSH respectively in terms of the net throughput. As expected, the net throughput under both heuristics tends to increase as the load inside the system becomes heavier as the inter arrival time gets smaller in value.

Comparing the two figures demonstrates the superiority of HoPe performance when net throughput is considered. An important observation is clear also where the net throughput of HoPe looks marginally affected by the grid scale. Consequently, the HoPe net throughput is mainly a function of the inter arrival time, which clearly demonstrates the efficiency of the dynamic role-altering technique adopted by HoPe, where the system virtualises the actual number of workers to cope with the current context requirements. In contrast, the OSH net throughput is significantly affected by the grid scale, particularly for low values of the inter arrival time. Mathematical equations and statistical data of the HoPe net throughput model and the OSH net throughput model are presented in Table 5 and Table 6 respectively.

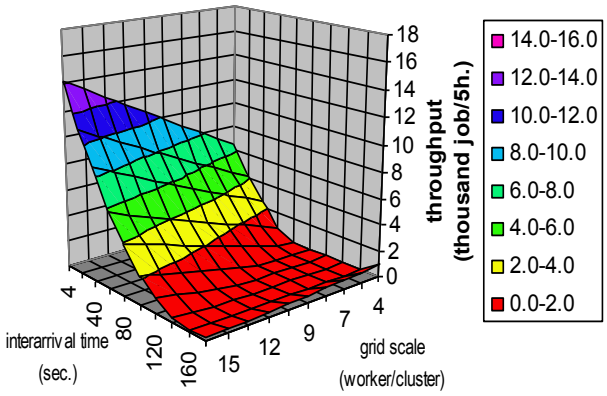

Fig. 7. HoPe throughput model illustrating high net throughput. An important observation is clear also, where the net throughput of HoPe looks marginally affected by the grid scale.

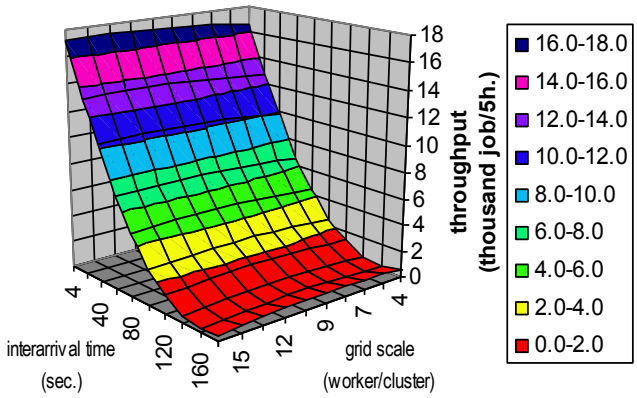

Fig. 8. OSH throughput model illustrating OSH net throughput is lower than HoPe and significantly affected by the grid scale, particularly for low values of the inter arrival time.

TABLE 5. Statistical Data of HoPe Throughput Model

HoPe_throughput $= b_0 + b_1 \times$ *inter arrival_time* $+ b_2 \times$ *grid_scale* $+ b_3$ *inter arrival_time* $\times$ *inter arrival_time* $+ b_4 \times$ *inter arrival_time* $\times$ *grid_scale* $+ b_5 \times$ *grid_scale* $\times$ *grid_scale*				
F = 7.332 *F-signif* = 0.000409				
Coefficients		***P-value***	**-95%**	**95%**
b_0	16666.6	0.03262	7880.13	25453.2
b_1	216.63	0.822	-1766.0	2199.3
b_2	-214.58	0.000334	-318.88	-110.28
b_3	-8.016	0.862	-102.51	86.47
b_4	-0.437	0.855	-5.370	4.495
b_5	0.685	0.00120	0.167	1.204

TABLE 6. Statistical Data of OSH Throughput Model

OSH_throughput $= b_0 + b_1 \times$ *inter arrival_time* $+ b_2 \times$ *grid_scale* $+ b_3$ *inter arrival_time* $\times$ *inter arrival_time* $+ b_4 \times$ *inter arrival_time* $\times$ *grid_scale* $+ b_5 \times$ *grid_scale* $\times$ *grid_scale*				
F = 10.44 *F-signif* = 3.87375E-05				
Coefficients		***P-value***	**-95%**	**95%**
b_0	4540.0	0.183	-639.418	9719.334
b_1	615.25	0.288	-553.455	783.954
b_2	-116.90	0.00101	-178.383	-55.416
b_3	2.33	0.569	-53.368	58.02841
b_4	-4.000	0.09841	-6.90689	-1.0913
b_5	0.565	0.00109	0.259235	0.870749

5.3 Turnaround Time (TT)

Figure 9 and Figure 10 summarise the behaviour of HoPe and the OSH respectively in terms of TT. The dominance of HoPe performance is clear by comparing the scales in the TT axis in the two figures. Figure 9 shows that the TT value under HoPe is gradually getting smaller as the grid becomes larger while the inter arrival time has a notably less effect in large grid scales. The case is different when it comes to the OSH, as illustrated in Figure 10 where the inter arrival time has an increased effect on the value of the TT.

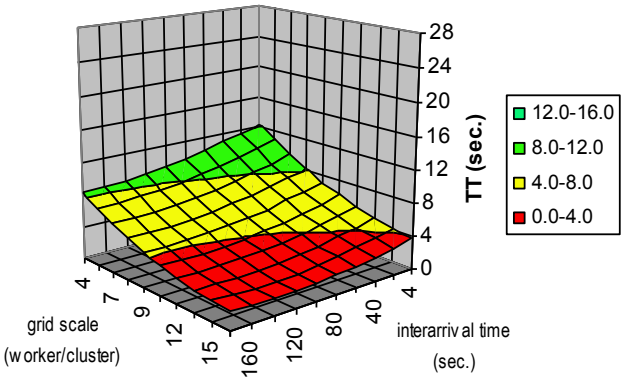

Fig. 9. HoPe TT model illustrating low TT value which is gradually getting smaller as the grid becomes larger while the inter arrival time has a notably less effect in large grid scales.

As expected, under both heuristics the TT approaches its minimal values as both the grid scale and the inter arrival time approach their maximum values, while the TT approaches its maximum as both approach their minimum. Mathematical equations and statistical data of the HoPe mean TT model and the OSH mean TT model are presented in Table 7 and Table 8 respectively.

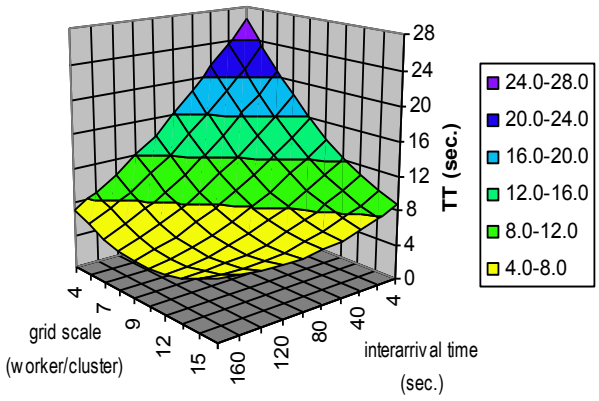

Fig. 10. OSH TT model illustrating low TT value is gradually getting smaller as the grid becomes larger while the inter arrival time has notably less effect in large grid scales.

TABLE 7. Statistical Data of HoPe TT Model

HoPe_ TT $= b_0 + b_1 \times$ *inter arrival_time* $+ b_2 \times$ *grid_scale* $+ b_3$ *inter arrival_time* $\times$ *inter arrival_time* $+ b_4 \times$ *inter arrival_time* $\times$ *grid_scale* $+ b_5 \times$ *grid_scale* $\times$ *grid_scale*				
F = 22.15 *F-signif* = 1.01124E-07				
Coefficients		***P-value***	**-95%**	**95%**
b_0	17.8	3.86E-10	15.26	20.34
b_1	-1.5	0.000172	-2.074	-0.926
b_2	-0.04678	0.00408	-0.07696	-0.01660
b_3	0.03958	0.00666	0.01224	0.06692
b_4	0.0011	0.03270	-0.000328	0.00253
b_5	0.00011	0.109	-3.94256E-05	0.000261

TABLE 8. Statistical Data of OSH TT Model

OSH_ TT $= b_0 + b_1 \times$ *inter arrival_time* $+ b_2 \times$ *grid_scale* $+ b_3$ *inter arrival_time* $\times$ *inter arrival_time* $+ b_4 \times$ *inter arrival_time* $\times$ *grid_scale* $+ b_5 \times$ *grid_scale* $\times$ *grid_scale*				
F = 32.67 *F-signif* = 3.19009E-09				
Coefficients		***P-value***	**-95%**	**95%**
b_0	37.80	3.34E-10	31.77	43.83
b_1	-3.000	0.000280	-4.361	-1.639
b_2	-0.185	2.57E-05	-0.256	-0.113
b_3	0.07492	0.02565	0.01005	0.140
b_4	0.00820	5.48E-05	0.00482	0.01159
b_5	0.000230	0.08247	-0.00012	0.000588

6 Related Work

A detailed survey on emerging grid systems is presented in [20]. The survey sheds the light on new paradigms in the area such as Personal Grids and Mobile Grids. In this section, we indicate how these systems are related to PM-Grids.

Connecting distrusted devices owned by an individual, or a group of individuals, and allowing them to share network resources is not the core of PM-Grids; PNs [4], PN Federation (PN-F) [16], Personal Grid (PG) [21][22] and Personal distributed Environment (PDE) [23][24] have been already proposed for this purpose. Allowing mobile access to grid systems is also not the core of PM-Grids; several projects, such

as [33] [25], have already addressed this issue. The novelty of PM-Grids are in superimposing computational grid functionalities on top of networked resource limited devices, whether they are mobile or stationary, and making the grid functionality available at personal users' hands. This section places PM-Grids amongst the above-mentioned projects and highlights the main similarities and differences. It is important to note that this paper focuses on the PM-Gird paradigm more than the resource scheduler HoPe [14][16] and this is why only related grid systems are reviewed in this section. Detailed survey of well-established grid schedulers, such as Condor [26], Legion [27] and Nimrod-G [28], and other bio-inspired scheduling algorithms, such as ant [29] and bee [30] is presented in [31].

6.1 PN and PN Federation

A PN offers a secure environment for a personal user to share network resources among his/her own devices. In MAGNET Beyond [4] and PNP2008 [32] the concept of PNs is extended into PN Federation (PN-F or Fednets), a secure cooperation between PNs of different users for a specific common purpose [20]. However, both PN and PN-F are concerned with sharing network resources such as data and peripherals rather than computing resources such as CPU cycles and runtime memories. Additionally, PN-Fs are formed only on demand for temporal situations; once the task is completed the network dissolves. On the other hand, PM-Grids are mainly concerned with sharing computing resources, and are set on a long-term basis for long-term goals.

6.2 Mobile Grids

The Akogrimo (Access to knowledge through the grid in mobile world) project [25] is the first IST project that explicitly targets Mobile Grids. While both Akogrimo and PM-Grid are concerned with integrating mobile devices in grid environments, Akogrimo is designed specifically for people in an enterprise domain, rather than for individual users in PM-Grids. The architecture of Akogrimo is based on an Enterprise Network, which is built out of a consortium of enterprises in contrast to a PN underlying a PM-Grid, which belongs to a single user. Additionally, mobile devices serve only as entry points to the grid in Akogrimo while they can participate actively in PM-Grids.

6.3 Personal Grids

A framework for a Personal Grid constructed over personal desktop computers is proposed in [22]. The framework consists of a two level hierarchal scheduling scheme where a super-node distributes jobs among clusters. Then, a master node in each cluster distributes the load among workers in FIFO style. The PM-Grid is different in that it extends the grid platform to mobile devices. Additionally, it has a distributed adaptive self-control scheduling scheme with no central entity, at the grid or cluster level, such as a super- or a central-node, making the scheduling decision.

The VEGA Grid project [21] has also proposed a framework for a Personal Grid (PG) to allow the integration of desktop computers into a "Global Grid System". In this platform, mobile devices are also used only as entry points to the grid. The PG aims primarily to establish a P2P platform for file sharing rather than processor sharing.

6.4 Personal Distributed Environment

In [23, 24] a Personal Distributed Environment (PDE) is proposed to allow a personal user to access his/her personal devices over heterogeneous networks to gain access to file sharing services such a global address book and the delivery of rich multimedia content. Again, the main concern here is data communication rather than computations.

7 Conclusion

The overall aim of the paper has been to introduce PM-Grids as a novel paradigm in grid computing for endowing individuals with resource-rich infrastructures that can serve as virtual general-purpose and mobile supercomputers. PM-Grids have the potential to bridge the gap between personal users and mobile devices on the one side, and current grid systems on the other.

The paper has also aimed to address the non-clairvoyant scheduling problem in grid computing, where job information is not available to the system before the end of the execution. HoPe, which is a novel honeybee- inspired resource scheduling heuristic with a decentralised self-management and adaptive scheduling policy, has been proposed to achieve this aim.

The paper aims have been fulfilled resulting in the following main contributions:

First, architectural designs and models for PM-Grids have been developed based on the PNs architecture and as a natural extension to them; an abstract layered view, a detailed inside view and simulated models have been presented and evaluated at different scales in terms of the numbers of jobs and devices per cluster.

Second, a detailed design, implementation and evaluation of HoPe have been initiated. To the best of our knowledge, HoPe is the first algorithm to shed light on the non-clairvoyant scheduling problem in grid computing. It is the first honeybee-inspired algorithm attempting to solve the resource scheduling problem relying totally on local and computationally simple parameters.

Third, a controlled empirical evaluation framework to prove the concept of PM-Grids and to evaluate the performance of HoPe has been developed. A flexible simulator has been built for this purpose allowing the control of experimental parameters, randomising extraneous variables as well as measuring and analysing various performance metrics.

Fourth, performance models of HoPe and OSH have been predicted in forms of mathematical equations and 3D graphical representations. These models are important to gain a clearer insight into the behaviour of each heuristic in regard to stability, net throughput, turnaround time and speedup under various running conditions of job inter arrival times and grid scales.

It can be concluded, based on the experimental results and predicted performance models, that using HoPe for resource scheduling in PM-Grids considerably reduced the effect of variations in grid scale and job inter arrival times, illustrating better scalability and sustainability, when compared to the OSH. However, these accomplishments need to be followed with thorough development efforts to transform the PM-Grid models into reality and apply HoPe in other contexts beyond PM-Grids. The work in this thesis opens up research on various

interesting issues and directions.

Acknowledgments

This work was partially funded by the Long-Term Comprehensive National Plan for Science, Technology and Innovation of the Kingdom of Saudi Arabia, grant number 11-INF1895-08.

References

[1] R. C.Braley, Ian C. Gifford, and Robert F. Heile, "Wireless personal area networks: an overview of the IEEE P802.15 working group," *SIGMOBILE Mobile Comput. Commun. Rev.*, 2000, vol. 4, pp. 26-33. http://dx.doi.org/10.1145/360449.360465

[2] S. A Mahmud, Kumendra Sivarajah, S. Khan and H. S. Al-Raweshidy, "Meshed High Data Rate Personal Area Networks," Journal of IEEE Communications Surveys & Tutorial , Vol 10, March 2008, pp. 58-69. http://dx.doi.org/10.1109/COMST.2008.4483670

[3] T. Suliman, K. Sivarajah and H. S. Al-Raweshidy," Modification of the IEEE 802.15.3 standardisation for PNC selection performance", IEEE Communications Magazine, vol. 45, Issue 12 pp.102-109, December 2007.

[4] IST.MAGNET Beyond (IST-FP6-IP-027369) [online]. Available: http://www.magnet.aau.dk, [accessed Nov. 2, 2013].

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

[6] M. Li and M. Baker, The Grid: Core Technologies. Wiley, 2005. http://dx.doi.org/10.1002/0470094192

[7] J.Kołodziej, S. Khan, L. Wang, M. Kisiel-Dorohinicki, S. A. Madani, E. Niewiadomska-Szynkiewicz, A. Y. Zomaya, C. Xu, "Security, energy, and performance-aware resource allocation mechanisms for computational grids," *Future Generation Computer Systems,* vol. 31, Feb. 2014, pp. 77–92. http://dx.doi.org/10.1016/j.future.2012.09.009

[8] R. Duan, R. Prodan, X. Li, "A sequential cooperative game theoretic approach to scheduling multiple large-scale applications in grids," *Future Generation Computer Systems*, Volume 30, Jan. 2014, Pages 27-43, ISSN 0167-739X

[9] A.J. Chakravarti, G. Baumgartner, and M. Lauria, "Self-organizing scheduling on the organic grid," *Int. J. High Performance Comput. Applicat.,* vol. 20, pp. 115–130, 2006. http://dx.doi.org/10.1177/1094342006061892

[10] J. Leung, L. Kelly and J. H. Anderson, *Handbook of Scheduling: Algorithms, Models, and Performance Analysis.* Boca Raton, FL: CRC Press, 2004.

[11] R. L. Graham, "Bounds on multiprocessing timing anomalies," *SIAM J. Appl. Math.*, vol. 17, no. 2, pp. 416-429, 1969. http://dx.doi.org/10.1137/0117039

[12] D.S. Johnson, A. Demers, J.D. Ullman, M.R. Garey and R.L. Graham, "Worst-case performance bounds for simple one-dimensional packing algorithms," *SIAM J. Comput.*, vol.3, pp. 299–325, 1974. http://dx.doi.org/10.1137/0203025

[13] K. Li, "An average-case analysis of online non-clairvoyant scheduling of independent parallel tasks," *J. Parallel Distrib. Comput.,* vol. 66, no. 5, pp. 617-625, May 2006. http://dx.doi.org/10.1016/j.jpdc.2005.06.007

[14] T. D. Seeley, *The Wisdom of the Hive: The Social Physiology of Honey Bee Colonies*. MA: Harvard University Press, 1995.

[15] H. Kurdi, "Method of forming a mobile grid system and resource scheduling thereon," patent- United States Patent and Trademark Office, US 8,296,765 B2; Issued October 23, 2012.

[16] H. Kurdi, M. Li and H. S. Al-Raweshidy, "A generic framework for resource scheduling in Personal Mobile Grids based on honeybee colony," in *Proc. of the IEEE 3rd International Conference on Next Generation Mobile Applications, Services and Technologies IEEE NGMAST '09*, pp. 297-302, Sep. 2008, Cardiff, UK.

[17] A. Legrand, M. Quinson, H. Casanova and K. Fujiwara, "The SIMGRID project simulation and deployment of distributed applications," in *Proc. 15th IEEE Int. Symp. High Performance Distrib. Comput.,* 2006, pp. 385-386.

[18] OPNET Technologies [online]. Available: http://www.opnet.com/, [accessed Dec. 11, 2013].

[19] E. Frachtenberg and D. G. Feitelson, "Pitfalls in parallel job scheduling evaluation," in *Proc. 11th Workshop Job Scheduling Strategies for Parallel Process.*, 2005, New-York, pp. 257-282. http://dx.doi.org/10.1007/11605300_13

[20] H. Kurdi, M. Li and H. S. Al-Raweshidy, "A classification of traditional and emerging grid systems," *IEEE Distributed Systems* [on line], vol. 9, no.3, pp.1, Mar. 2008. ISSN: 1541-4922 .

[20] M. Ibrohimovna, and S. H. Groot, "Proxy-based Fednets for sharing personal services in distributed environments," *in Proc. 4th ICWMC*, 2008, pp.150-157.

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

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

[23] D. Pearce, J. Dunlop and R.C. Atkinson, "Leader election in a personal distributed environment," in *Proc. IEEE 16th Int. PIMRC*, 2005, vol. 2, pp. 1307-1311.

[24] J. Dunlop, "The concept of a personal distributed environment," *Wireless Personal Commun. Int. J.*, vol. 42, no. 3, pp. 431-444, 2007. http://dx.doi.org/10.1007/s11277-006-9186-7

[25] Akogrimo [online]. Available: http://www.mobilegrids.org/, [accessed Dec. 11, 2013].

[26]Condor Project [online]. Available: http://www.cs.wisc.edu/condor, [accessed Dec. 11, 2013].

[27] Legion: A Worldwide Virtual Computer [online]. Available: http://legion.virginia.edu/, [accessed Dec. 11, 2013].

[28] DSTC Nimrod/G [online]. Available: http://www.csse.monash.edu/~sgaric/nimrod/, [accessed Dec.

11, 2013].

[29] R.Chang, J.Chang, P. Lin, "An ant algorithm for balanced job scheduling in grids", Future Generation Computer Systems 25, pp. 20–27, 2009. http://dx.doi.org/10.1016/j.future.2008.06.004

[30] J. Taheri, Y. Lee, A. Y. Zomaya, H.Siegel, "A Bee Colony based optimization approach for simultaneous job scheduling and data replication in grid environments," Computers & Operations Research, Volume 40, Issue 6, June 2013, Pages 1564-1578, ISSN 0305-0548,

[31] H. Kurdi, "Personal Mobile Grids with a Honeybee Inspired Resource Schedule," PhD thesis, School of Engineering and Design, Brunel University, Uxbridge, UK, 2010.

[32] The Dutch Freeband Communications Project PNP2008 [online]. Available: http://www.utwente.nl/ctit/research/projects/concluded/bsik/freeband/projects/pnp/, [accessed Dec. 11, 2013].

[33] j. Paul, Darby III, N. Tzeng, "Decentralized QoS-Aware Checkpointing Arrangement in Mobile Grid Computing," *IEEE Transactions on Mobile Computing*, vol. 9, no. 8, pp. 1173-1186, Aug. 2010, doi:10.1109/TMC.2010.80 http://dx.doi.org/10.1109/TMC.2010.80

[34] Valderi R. Q. Leithardt, David Nunes, Anubis G. M. Rossetto, Carlos O. Rolim, Cláudio F. R. Geyer, Jorge Sá Silva, Privacy Management Solution in Ubiquitous Environments Using Percontrol, Journal of Ubiquitous Systems and Pervasive Networks, Volume 5, Issue 2, pp. 21-28, 2014. http://dx.doi.org/10.5383/juspn.05.02.004

[35] Christina Strohrmann, Julia Seiter, Gerhard Tröster, Feedback Provision on Running Technique with a Smartphone, Journal of Ubiquitous Systems and Pervasive Networks, Volume 5, Issue 1, pp. 25-31, 2014. http://dx.doi.org/10.5383/juspn.05.01.004