Pervasive evolutionary algorithms on mobile devices*

P. Garcia-Sanchez, J.P. Sevilla, J.J. Merelo, A.M. Mora, P.A. Castillo, J.L.J. Laredo, and F. Casado

Department of Computer Architecture and Computer Technology, CITIC-UGR, University of Granada, Spain
{pgarcia,jmerelo}@geneura.ugr.es

Abstract. This paper presents a Java framework to implement distributed applications via Bluetooth. It provides a high-level Application Programming Interface (API) which simplifies the creation of applications for Bluetooth devices in Java ME and Java SE platforms. This framework is based in a client-server architecture and an event-driven asynchronous communication mechanism. As an example of use, we solve two well-known evolutionary computation problems (the Traveler Salesman Problem and the Wave Function Problem).

1 Introduction

It is a well-known fact that the mobile technology is increasingly present in our society. It is strongly associated to the new communication technologies that offer to users a whole set of possibilities. These devices have a computation power that, in general, is being wasted, but with the right tools this power could be used to solve complex computational problems.

One of these emerging technologies is Bluetooth [1]. It lets wireless communication between mobile devices, but just inside an limited area (about 100 metres radio). The percentage of terminals that include that technology is growing, but its real possibilities are not being exploited at all, and the main activity performed is only the interchange of files, they not using its real "interactive" capabilities.

The goal of this work is provide a tool to simplify the creation of scalable communication applications, based on a layered design for standardization and re-utilization of the developing process of an application [2]. These applications can be of any kind, including chats, video-games, remote-controlling of other devices (like a PC) or even the usage of mobile devices in distributed programming, as the case we are presenting.

Specifically, Evolutionary algorithms (EAs) are well suited for distributed computing, being this kind of algorithms perfectly adapted to this technique, due to the easily parallelization of the algorithm execution in several kind of networks.

* Supported by projects AmIVital (CENIT2007-1010) and EvOrq (TIC-3903)

The rest of the work is structured as follows: first the state of the art in existent Java applications is described (section 2). Section 3 introduces the technologies used in the development of this work. After this we present in section 4 the design of the proposed architecture (called Ulfsark) and the development of a distributed computing application using a genetic algorithm with the experiments and yielded results (in section 5). Finally the conclusions and future work are exposed.

2 State of the Art

Pervasive computing in mobile devices is a new flexible and extensible way to delegate tasks dynamically to ad-hoc wireless networks, being the reliability of the network an important issue to investigate [3]. For example, mobile devices have been used to analyze data by specialists, receiving data and giving human feedback in medical environments [4], using the SOAP protocol for the data transmission.

The work by Cano et al. in [5] is of interest since they demonstrate the feasibility of Bluetooth in ad-hoc networks. It consists of a framework for data interchange in P2P networks. Nevertheless, the experiments in real environments have lower performance than in Bluetooth simulators, but they demonstrate the feasibility of use Bluetooth for the deployment of applications and communications in spontaneous networks. The power of this mobile networks also can be used to implement distributed databases, also using Bluetooth for to access the data in a ad-hoc wireless network [6].

Nowadays there are a lot of commercial applications which apply Bluetooth communication in mobile devices. Many of them are company-specific, so they use proprietary and inaccessible languages. Other applications are implemented in C++, being operable in a little set of devices, like BuzzZone [1], that lets the contacts search and communication. In addition, there are some Java-based projects, being the main reference MobiLuck[2], used to create meetings among friends.

Simultaneously to the development of this work several open-source Java-based projects have emerged, like Valhalla[3], but it is oriented to a simple chat. Regarding distributed computing using mobile devices it is interesting to show the Boincoid Project[4]. It runs over Android SO devices, and allows to donate idle time to be used in scientific projects, following the Boinc [7] philosophy. However the Android devices are not extended (only a unique model can be purchased).

Relating the convergence between Bluetooth and GAs, Screenivas and Ali proprose in [8] the use of Genetic Algorithms (GAs) to create Bluetooth networks (called *scatternets*).

[1] `http://www.buzzone.net/eng/technologies.html`
[2] `http://www.mobiluck.com/`
[3] `http://www.valhallachat.com/`
[4] `http://boincoid.sourceforge.net/index.html`

3 Used technologies

This section details the tools and communication protocols employed within the presented framework.

Bluetooth is the specification that defines a wireless global communication standard for data and voice transmission between different devices through a radio frequency link [1, 9]. Bluetooth main issues are: to facilitate the communications between mobile and fixed devices, remove the wires and connectors and allow the creation of small wireless networks to let the synchronization of data. The optimum communication range for two devices is 10 metres (reaching to 100) and the transmission speed is 720Kb/s.

Bluetooth protocol stack is composed by specific protocols (as L2CAP, RFCOMM or SDP[1]) and adapted protocols (like OBEX [10]). The L2CAP layer is the core of the stack and includes packet segmentation and data reordering. The RFCOMM layer provides emulation of multiple RS-232 serial ports between two Bluetooth devices. A device can have more than one RFCOMM session with some different devices (one session per device); it can connect with seven different devices at the same time.

When two or more Bluetooth devices inside the range establish a connection, a Personal Area Network (PAN) is created. This network can be a *piconet* or a *scatternet*. The first one has only one master and up to seven slaves, being the master the device which starts the connection. If an eighth device wish to connect to a *piconet* network the master could not add it until one of the existent slaves leaves the network. But if one of the slaves support multi-point connection, then the new device can connect to these device, creating a *scatternet* (see Figure 1).

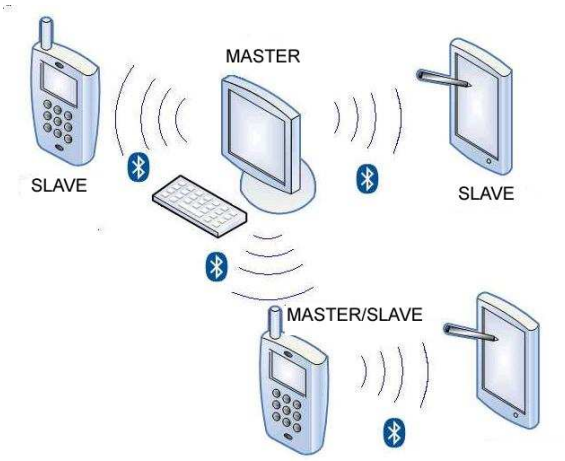

Fig. 1. Several devices connected in a scatternet

The application has been developed in Java to implement our application creation since it requires a lower development cost and because is platform-independent. Moreover, it is the only language with an standardized Bluetooth API[11]. We have used Java ME[12], that is an API collection oriented to the development of software for low-resources devices, like PDAs, mobile phones and others.

Distributed computing lets to make the most of parallel processing (in order to increase the computing performance) with a lower cost than other multiprocessor architectures, because it is based on the use of standard hardware elements, which have a wide market, and where is more easy to repercute the research and development costs to produce the best performance elements. Two clear examples are the research lines centred in clusters [13] and GRID [14] for parallel processing.

We want to take advantage of this idea and apply it to mobile devices environment. The increasing performance of these devices is transforming them into great computers and we should be able to exploit their resources. The required elements for the distributed parallel programming are the computer devices and the communication environment; in our case they will be the mobile phones and Bluetooth, respectively.

On the other hand, we have considered the evolutionary computing, which are able to solve complex problems but they need a lot of computation resources. In this way, the distributed computation is a very attractive technique to increase the performance of the genetic algorithms (GAs) [15], (the most used of evolutionary algorithms).

Several implementations of evolutionary parallel algorithms have been proposed [16] being the most used:

- *Global population model*: one population is created and the evolutionary operators application is performed parallely.
- *Island Model or coarse-grain*: individuals are distributed among different subpopulations assigned to several processors, which locally apply the evolutionary operators to their population and exchange solutions using a migration policy, so the populations evolve in a relatively independent way.
- *Cellular model or fine-grain*: population is divided into many subpopulations with a few individuals (usually just one), and assigned to each processor. The selection and crossover operations are performed between individuals of neighbour processors.

4 Ulfsark framework

This section dives in the functionality and design of the proposed framework, called Ulfsark. Ulfsark is a framework to create Bluetooth-based applications. The main issue to solve is to abstract some details of the API specified in the JSR-82 document, and to offer to programmers some key features:

- Simple interface

- Send/Receive asynchronous data transference
- Package delimited flow
- Event oriented programming
- Client-server model

The source code is available at `http://ulfsark.sourceforge.net`.

As previously said, one of the Ulfsark objectives is to constitute an intermediate layer between the Bluetooth API and any application, and also to automatize the parallel reading of the received packages and event generation.

The client-server model establishes a central entity which creates the services and manages the connected clients. So, a *client* could find the *server* services using a *searcher*, and could connect to every available service. The server decides which petitions accepts and distribute messages between clients.

The system has been designed in three layers. The first one defines data packages and an abstract server and client interface. The second one depends on the application and it adds functionalities to server, clients and packages. Finally, the last is the presentation layer, which is adapted to the device features. An example is showed in Figure 2.

Fig. 2. Two applications using Ulfsark architecture: a chat, and a distributed computing application

Instead of a server-centralized architecture, the all-to-all architecture of Ulfsark reduces the overload of the message management by avoiding message queue and transmision to the server.

The way that our application has been used for distributed computing is as follows: when one of the mobile devices executes the application it activates its server with a specific service number (UUID, the same for all Ulfsark applications), and search for active services with the same UUID. Once this search is finished, the device creates as many clients as services found, and it connects with all servers with the specific UUID. At the same time, every device server receives the connection requests of all clients, so it handles all connected devices.

Finally, all devices are connected in an all-to-all network and every device waits for the events that may arise.

During the algorithm execution, every mobile device sends its best individual every pre-defined number of iterations to another randomly selected device, following the previously explained island model. This individual is added to the actual population of the receiving device in the next generation. When a device finish its execution it sends a termination message to the others containing the best individual of its population.

5 Experiments

To test Ulfsark two genetic algorithms to solve well-known problems have been implemented:

- **Wave function:** The problem consists in solving the (x,y) values that maximize the function 1.

$$f(x,y) = \frac{sin(\sqrt{x^2+y^2})}{\sqrt{x^2+y^2}} \quad (1)$$

  This function is multi-modal, that is, there exist many local optima where an analytic method would be stagnated, not reaching the global optima in the (0,0) coordinate, where $f(x,y) = 1$. Chromosomes are in this case a vector with two coordinates. The crossover simply cross the two genes and the mutation consists in adding or subtracting a very small random number to one of the components.
- **Travelling Salesman Problem:** This problem consists in find the optimal route, that beginning and ending in the same city, and visiting all cities minimize the distance travelled by the salesman. Here the individuals are vectors that represent a list of cities and the fitness function (to minimize) is the travelled distance following this order. Crossover interchanges a segment of the tour and the mutation swaps two cities in the vector [17].

The chosen mobile device is the Nokia 6288 mobile phone, with a 15.7 Mhz Java virtual processor speed and a 2048 KB RAM memory. The parameters of the two algorithms are shown in Table 1. Every experiment was run 15 times and the results are shown in Table 2.

It can be seen that the distributed version uses less generations to find the best individual, being the number of individuals and maximum generations the half of the sequential version, because the island model have effect on the behaviour of the algorithm and the populations does not converge to the same solution.

6 Conclusions and future work

We have presented a distributed genetic algorithm implementation using Bluetooth in mobile devices to solve two problems, and we have demonstrated that

Table 1. Parameters of the experiments. For each problem (TSP and Wave) there are two sequential versions (Seq1 and Seq2), the second considering the double of generations, but with the half population size. The distributed version (Dist) uses the minimum population size and number of generation of the previous ones.

Parameter	TSPSec1	TSPSec2	TSPDist	WaveSec1	WaveSec2	WaveDist
Individual size	10	10	10	2	2	2
Population size	16	8	8	100	50	50
Crossover probability	70	70	70	70	70	70
Mutation probability	80	80	80	80	80	80
Generation Number	500	1000	500	200	400	200
Tournament Selection	2	2	2	40	40	40

Table 2. Experiment results (average ± standard deviation). It can be seen that in the distributed version the generation to find the best individual and its fitness are better (lower in TSP and higher in Wave) than in the sequential versions, sacrificing the execution time, which is increased due to the Bluetooth communication latency.

Experiments	Fitness	Generations	Time to best (ms)	Total time (ms)
TSPSec1	24.53 ± 1.81	180.73 ± 90.32	1368.33 ± 731.44	3517.2 ± 1171.07
TSPSec2	25.2 ± 1.66	163.6 ± 148.92	1151.07 ± 1074.76	6910.13 ± 2576.32
TSPDist	24.4 ± 1.55	77.73 ± 57.03	6787.2 ± 1656.88	10539.33 ± 1559.68
WaveSec1	0.9999997131 ± 6.737e-7	162.2 ± 31.68	10849.53 ± 2649.23	13599.4 ± 1395.95
WaveSec2	0.9999998548 ± 2.788e-7	336.67 ± 57.07	9720.2 ± 1571.13	11544.73 ± 411.52
WaveDist	0.9999998874 ± 1.841e-7	79.27 ± 43.38	14853.64 ± 3307.52	20420.55 ± 3309.99

better solutions are found in less generations relating to sequential implementation, but sacrificing execution time due to Bluetooth latency.

Distributed computing in mobile devices it is not being used as much as would be desirable, taking into account that these devices have almost the same capabilities that computers in previous years. Moreover, the number of these devices is higher than personal computers and they include very simple communication methods, so they can create networks anywhere using technologies like (Bluetooth for instance), not widely used for computation. This work presents a simple and usable framework for the development of Bluetooth-based applications. It has been implemented in Java because the number of compatible devices is higher than any other language. As future developments of the proposed tool we are studying the next applications:

- Polling system
- Multiplayer Games
- Teaching uses
- Chats in leisure places
- PC management from mobile devices

Furthermore, we think that the improvements in P2P Evolutionary Computing can be key to develop more suitable models to distributed GAs in mobile networks. Both environments share a good amount of common issues such as decentralization, asynchrony, heterogeneity or unreliability.

References

1. SIG, B.: Bluetooth specification. `http://www.bluetooth.org/spec/` (2004)
2. Larman, C.: Applying UML and Patterns. Prentice-Hall (1998)
3. Ferrante, A., Pompei, R., Stulova, A., Taddeo, A.V.: A protocol for pervasive distributed computing reliability. In: Proceedings of the 4th IEEE International Conference on Wireless and Mobile Computing, Networking and Communication, (WiMob 2008). (2008) 574–579
4. Correa, B., Ishikawa, E., Ziviani, A., Faria, M.: Medical image analysis using mobile devices. In: Proceedings of the ACM Symposium on Applied Computing. (2008) 1380–1384
5. Cano, J., Cano, J., Manzoni, P., Kim, D.: On the design of pervasive computing applications based on Bluetooth and a P2P concept. In: Proceedings of 1st International Symposium on Wireless Pervasive Computing. (2006) 1–6
6. Artail, H., Shihab, M., Safa, H.: A distributed mobile database implementation on Pocket PC mobile devices communicating over bluetooth. Journal of Network and Computer Applications **32**(1) (2009) 96–115
7. Anderson, D.P.: BOINC: A system for public-resource computing and storage. (2004) 4–10
8. Sreenivas, H., Ali, H.: An evolutionary Bluetooth scatternet formation protocol. In: Proceedings of the Hawaii International Conference on System Sciences. Volume 37. (2004) 4893–4900
9. Wei, P., Chen, C., Chen, C., Lee, J.: Support and optimization of Java RMI over a Bluetooth environment. Concurrency Computation Practice and Experience **17**(7-8) (2005) 967–989
10. IrDA: OBEX specification. (`http://irda.org/`)
11. Hopkins, B., Antony, R.: Bluetooth For Java. Apress! (2004)
12. Microsystems, S.: Java ME specification. (`http://java.sun.com/javame/reference/apis.jsp`)
13. Buyya, R.: High Performance Cluster Computing: Architectures and Systems. Prentice-Hall (1999)
14. Foster, I.: The Grid: A new infrastructure for 21st Century Science. Phisics Today **55** (2002) 42–47
15. Alba, I., Tomassini, M.: Paralellism and evolutionary algorithms. IEEE Transactions on Evolutionary Computation **6** (2002) 443–462
16. Chambers, L.: Practical Handbook of Genetic Algorithms: Complex Coding Systems. CRC-Press (1998)
17. Larran aga, P., Kuijpers, C.M.H., Murga, R.H., Inza, I., Dizdarevic, S.: Genetic algorithms for the travelling salesman problem: A review of representations and operators. Artificial Intelligence Review **13**(2) (1999) 129–170