Pocket Caching: A Strategy of Prefetching Cache Based on Multi-objective Optimization for Mobile Cloud Computing

María del Pilar Villamil G. and Carlos Javier Urango M.

***Abstract*—This paper presents pocket caching, a prefetching cache strategy to identify the data to be loaded in a mobile node participating in a mobile cloud computing (MCC) environment. MCC is a new paradigm and is consider as the extension of mobile devices services in an elastic and transparent way through the ubiquitous wireless access to third-party computing and storage resources. New applications are emerging in this context. They are mainly characterized by large amount content generation in a collaborative and dynamic way. Additionally, the user interaction (content access and generation), depends on user location and interaction of others users. User interaction may be affected by network problems in some points of interest. This motivate pocket caching proposal. Pocket caching to reduce data availability issues in points of interest, designs a multi-objective optimization strategy to include characteristics of MCC environment and some application behaviors. This proposal was validated using both the SPEA2 algorithm execution performance and the quality of the results. The preliminary results show good behavior.**

***Index Terms*—Mobile cloud computing, multi-objective optimization, prefetching cache.**

I. Introduction

Mobile devices, like smartphone or tablet, have high degree of participation in daily lives of people. According to [1] in 2017 traffic mobile and wireless devices will represent 55% of IP traffic. These devices have been growing in terms of processing power, storage and memory, and they are in a continuous technological innovation. Additionally, they provide a richer user experience, enabling the execution of a wide variety of mobile applications over 3G or 4G wireless connections.

At the same time, a significant advance in the Cloud Computing paradigm is being observed. There are an increasing number of services that are being delivered, in a ubiquitously way, to different types of users. Cloud allows users to pay only for the computational resources they use and provides universal access to infrastructure, platform and software services. Specifically, storage and processing are services provided to improve functionality and performance of applications. The use of Cloud in a mobile environment has become very attractive. Mobile applications using Cloud services creates the Mobile Cloud Computing paradigm (hereinafter MCC). Although there is no consensus in the MCC definition, this can generally be seen as improving mobile devices using cloud services with connections wireless network.

A study by Juniper Research [2] highlights an annual increase of 88% from 2009 to 2014 in mobile software based on the Cloud. It shows the emergence of a new variety of mobile applications using intensive computing tasks. Applications such as image and text processing, content filtering, sensor data sharing, multimedia search can now be realized. These applications demand high processing and data storage capabilities that cannot be met only by making more powerful mobile devices.

As a consequence, MCC efforts must now focus on creating an infrastructure to give explicit support to this new style of applications. There are many applications, someone inspired by Web 2.0, that can be executed in a MCC infrastructure.

Sotamaa in [3] and Zhu in [4] propose a multiuser fights game that occurs in a virtual map. The scenario is built from a real map and the physical infrastructure (e.g., streets and buildings of the city). In the game users must locate, fight and destroy other players. This way they can earn credits and reputation in the community. Players are organized in clans and fights arise between players from different clans. The clans have security points located at different coordinates in the virtual map. These points provide improvements in the fight profile of players (e.g., new weapon or new shield). Besides, players can leave support messages associated to the points, which are then viewed by other players of the same clan (e.g., warning of danger).

The aforementioned applications are mainly characterized by large amount content generation in a collaborative and dynamic way (i.e., between users of the same clan). Additionally, the user interaction (content access and generation), depends on user location.

The execution of this kind of applications in a MCC environment can be affected by the constraints in the mobile devices resources related to storage, processing and battery. Energy, for example, is the limitation with greatest impact on the mobile application performance, because it is necessary to complete computational tasks. In the same way, the user mobility also affects the application performance, mainly because the device location changes produce variations e interruptions in the network connection status.

On the other hand, associated costs with the mobile network service produced by using these types of applications for long periods of time should be taken into account. This might become too much expensive for the

Manuscript received September 18, 2014; revised January 20, 2015.

The authors are with the Department of Systems and Computing Engineering, University of Los Andes, Bogotá, Colombia (e-mail: mavillam@uniandes.edu.com).

DOI: 10.7763/IJCTE.2016.V8.1026

user. Finally, the traditional Cloud was designed primarily to support business applications. It does not guarantee the availability of their computing services in a mobile environment.

The above, evidence new challenges related to the user experience, performance applications in a mobile environment, efficient resources use (i.e., mobile device, network and cloud computing), and efficient data access.

This work focus on a data availability problem related to applications where user interaction depends on user location and requires constant transfer of multimedia content between mobile devices and the Cloud (hereinafter MCC location-based applications). In this style of application, when a user acquires certain location, for example, it could maintain a wireless connection with limited bandwidth. (e.g., 2G to 90 kbps) affecting the access of data stored in the cloud, because the user may experience intermittent connectivity (e.g., While the user is riding as a passenger in a car), or simply may not have internet connection (e.g., area without network coverage).

Such situations together with user mobility result in a data availability problem. Devices may not always have available the data stored in the Cloud.

There are some works in the literature such as [5], [6] and [7] related to these issues. They propose Cloud services to create an efficient data storage structure to improve data access using user behavior patterns, and, some of them [6] and [7] based on cache. Although, these proposals are focus on provide timely data and reduce the energy consumption of the device, they do not have a data access mechanism adapted to a MCC runtime environment.

This was the main motivation to proposes the Pocket caching strategy, a prefetching cache mechanism that leverages local resources of mobile devices and the points where the device have good network capabilities to increase data availability (naming in the following Z points). Specifically, this work focuses on an optimization model to define the data to be loaded in Z points. This model takes into account mobile device, network connection and Cloud constraints together with the application characteristics. The first results obtained using the model evidenced that the proposed mechanism can be a valid approach to adapt the access process to Cloud data in a MCC context. The first results obtained using the SPEA2 algorithm shows good properties of the model and evidences a valid approach to improve data acquisition from the Cloud in a MCC context.

The rest of this paper is organized as follows: Section II presents a MCC background. Section III presents different works related to data availability issues in a MCC context. Section IV presents an overview of Pocket Caching proposal. Section V presents the problem and solution strategy of selection of data stored in a Cloud for Pocket Caching strategy. Section VI presents the system evaluation, and finally, Section VII presents the conclusions and future work.

II. Mobile Cloud Computing

There is no consensus about Mobile Cloud Computing (MCC) definition. This section gives a background in MCC in terms of a definition, and architectures.

A. Definition

MCC is conceived as a combination of mobile computing and cloud computing. Basically, it consists of 3 main technologies: Mobile, Cloud and Ubiquitous Computing. The first one is made up of three main elements: Mobile devices and communication components (Hardware), applications running on mobile devices (Software), and mobile network infrastructure, protocols and data delivery mechanisms (Communication). The second one is an on-demand provisioning infrastructure, platform and software as a service to the user from any device with Internet connection. Finally, the last one is defined by Weiser in [8] as a method to enhance computer use by making many computers available in the physical environment transparently to the user.

The combination of these three technologies provides new capabilities for mobile devices resources, which can be exploited by mobile applications. However, the problems and constraints of Mobile Computing (e.g., Connectivity level, data security and resource constraints) and Cloud Computing world (e.g., Privacy and data ownership) can affect the execution performance of the applications. Additionally, it is important to remember that the services offered in traditional cloud were not designed to support changes in the connectivity due to user mobility. As a result, the initiative to create changes in the MCC infrastructure to give explicit support to this new style of applications appears. Propose changes should focus on issues such as mobility devices, latency values fluctuating, energy saving, data availability, among others.

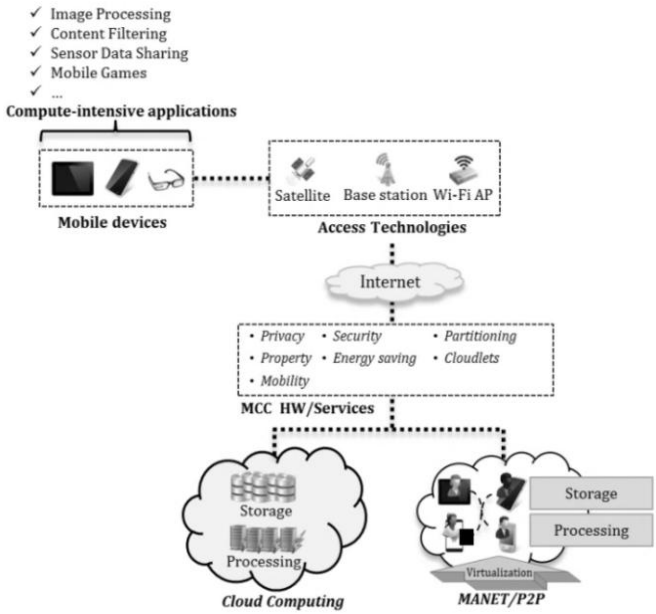

Fig. 1. MCC architecture definition.

In order to understand better the MCC context, this paper proposes a definition to MCC, which is illustrated in Fig. 1.

MCC is a paradigm aimed at enhanced the mobile devices capabilities in elastic and transparent way through the ubiquitous wireless access to third-party computing and storage resources. This paradigm enables to extend the Cloud infrastructure to give explicit support to mobile applications. In this way, it is possible to delegate tasks to third-party computing resources provider (e.g., Amazon

Web Services or MANET mobile network).

B. MCC Architecture Approaches

Three different approaches to MCC architecture have been proposed in the literature. The first approach is a resource-limited mobile device running a service hosted on a server in the cloud. The device acts as a thin client communicating with the server through a mobile connection (e.g., 3G).

The second approach suggests that mobile devices take advantage of computational capabilities of the Cloud through offloading of tasks and data. The offloading is defined so that fits the variable execution environment present in MCC. This is achieved by finding a balance between local execution in the device and remote execution in the Cloud (e.g., cost/benefit analysis).

Finally, the third approach refers to implement a mobile application using computational resources provided by a mobile cloud. The mobile cloud will consist of a number of mobile devices in the proximity of a user having connection problems to remote servers from traditional cloud.

The work in this paper is targeted to the first MCC architecture approach, although the general proposal is posed considering a hybrid development mixing the three approaches.

III. Related Works on Data Availability Issues

This section presents two groups of proposals that provide a basis for analysis of the challenges related to data access issues. The aim of these proposals is the creation of new services deployed in the cloud and on mobile devices. In this way, it gives an explicit support to intensive resource MCC applications through the use of cloud resources.

A. Sensor Data Streams

Fakoor *et al*. in [9] and Sen *et al*. in [10] propose the use of a mobile device as a sensor data stream generator node. The data generated is sent to the Cloud and processed for integration. These works propose to use sensors data as a service to applications based on the user context. Additionally, Sen *et al*. in [10] creates an interface for queries processing over data streams. The processing is made by a special language designed to detect collective context information about users. These proposals improve the computational capabilities of devices while reducing power consumption, taking into account the energy and storage restrictions on mobile devices.

B. Data Access

Shen *et al.* in [5], Koukoumidis *et al.* in [6] and Vemulapalli *et al.* in [7] focus on the storage and access of multimedia data that no change in a frequent way. They use Cloud services to create an efficient data storage structure. Data organization is made based on the user behavior patterns. In Shen *et al.* [5] case, the data organization is based on the user needs, identified using social media data. Koukoumidis *et al.* in [6] and Vemulapalli *et al.* in [7] proposals are based on cache mechanisms.

In Koukoumidis *et al.* the data stored in the cache is the more frequent data accessed by users from the Cloud. These data is identified using user and community models to detect patterns of data access. In Vemulapalli *et al.*, case, the data stored in the cache is selected according to the user behavior in terms of query frequency and the number of requests at each location. These proposals try to provide data access in an appropriate way, while power consumption is reduced. They take into account the energy restrictions and the mobile network connections characteristics.

In general, mobility can produce connectivity problems to remote servers in the Cloud. Furthermore, different connectivity levels produce different energy consumption on mobile devices. These issues affect the quality service of mobile applications executed in a MCC environment. As a consequence, MCC services and hardware, and heterogeneous access technologies are required to tackle the variability of the execution environment.

IV. Pocket Caching

Pocket Caching, as seen in Fig. 2, is a distributed prefetching cache service between Cloud and mobile devices (hereinafter mobile nodes). Both the Cloud and each mobile node can store static data about the application configuration, and also, dynamic data generated by users during the application interaction.

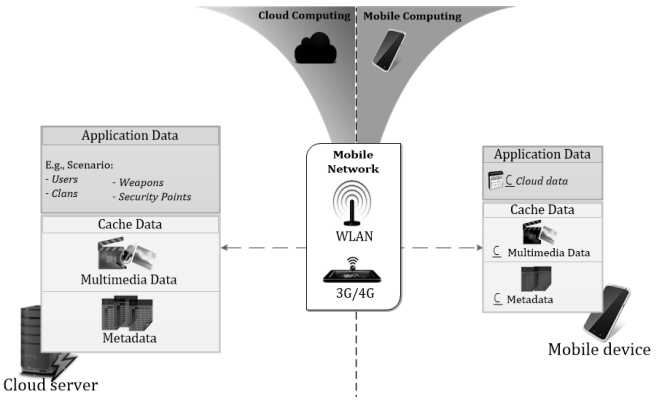

Fig. 2. Pocket Caching cache.

The cache memory on the mobile nodes stores relevant information to the application process before it is required (in a preload way). The target application style analyzed in the strategy proposed is described below, including the relationship to pocket caching.

A. MCC Location-Based Applications

The applications tackle in this paper are collaborative multi-user mobile applications using Cloud services to meet scalability and on demand data access requirements. In this type of applications users create and share multimedia content through mobile devices. User interaction with the application depends on the location and the information generated by others users.

In MCC location-based applications, a data request is resolved according to node's location. In this way, pocket caching chooses nodes with good network capacity, named the *elected nodes* in the following, to obtain in advance the multimedia content associated with future nodes interaction. These interactions occur in nodes having a high probability of presenting connectivity problems. These nodes are named *fragile nodes*.

B. System Model

Pocket Caching proposal take into account different cache management issues in MCC location-based applications. In particular, this preliminary version focuses on challenges related to elected node identification and the selection of data to be preloaded in the local cache of a specific node.

This paper focuses on issues related to the selection of data to be preloaded in the local cache of a mobile node, taking into account environment restrictions, object characteristics and user behavior.

Prediction of user path: Some works in the literature such as Henao in [11], Hariharan in [12], Liao in [13], Kang in [14] and Kjærgaard in [15] give ideas to identify user path. They propose mechanisms to detect patterns of user behavior based on places (e.g. Workplace, travel, and home) and time periods (e.g., day, week or month). They suppose similar behaviors between user groups according to user interests and preferences. Works such as Liao, Kang and Kjærgaard propose the use of different types of devices like GPS, radio frequency, accelerometer or compass to collect user movements and take information to identify user path. Although, this problem has been tackle by different authors, these proposals are not adapted to MCC context, and are considered as future work.

Classification of nodes and identification of elected nodes: These steps can be resolved using data mining techniques such as clustering. The main challenges may be related to the latency. This paper supposes the use of data about network characteristics; time spent in a node or in a near zone close to it, and proximity nodes with high probability to have communication problems. These data is used as criteria to identify elected nodes. Finally, the *selection of data to be preloaded* is presented in the next section.

V. Selection of Data to be Preloaded

Once, a user is on a Z point (point l1), and a F point (point l2) associated to point S1 is identified, it is possible to decide the data to be preloaded in the mobile cache. This data is related to point l2, and allows the user to maintain a minimum level of interactivity with the application. The decision about the data to be loaded in the mobile device is a complex problem, because of the large volume of information that can be associated with the point, generated by a large number of dynamic users participating in the application. This section presents the strategy proposed based on multi-objective optimization to resolve this issue.

A. Global View

The selection of data to be preloaded will be resolved in a MCC context. As we can see in Fig. 3 the decision depends on time constraints (user stay time in the location), local resources constraints (storage and energy of the mobile), the network connection status and the data characteristics in the Cloud, such as relevance and size. One hypothesis in this work is that all data in Cloud has the same relevance to the user.

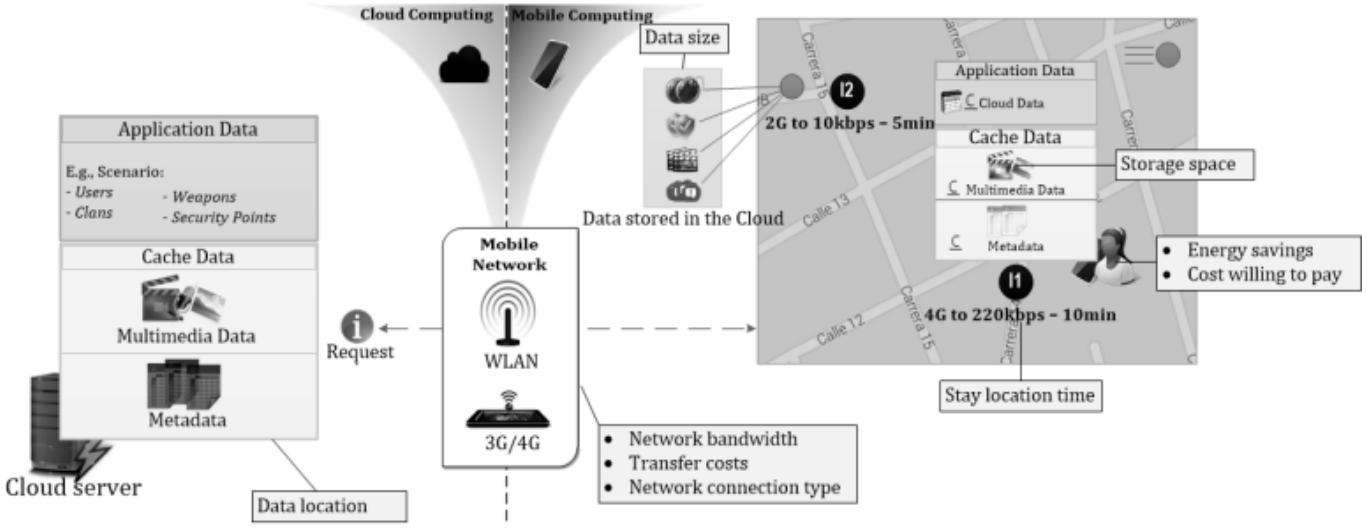

Fig. 3. Selection data to preload problem.

According to local resources, the energy consumption in the mobile should be used efficiently. As a result, the object identification process in the Cloud and the downloaded objects to mobile will use energy without it becomes fully exhausted. A second restriction is related to the monetary cost generated in the mobile network operator during transfer and receipt of data. This value depends on network connection type available during the request, and should also be minimized. Excessive monetary costs may occur for users running long time this application style.

Additionally, the volume of data to be downloaded must fit in the local cache, and the download time of the selected data and the query processing time should not exceed the time spent by a user on the location point.

Finally, the object selection process is seen as a multi-objective optimization problem with constraints and energy costs as sub-objectives of the problem.

B. Multi-objetive Optimization Problem

According to Donoso in [16], the multi-objective optimization refers to have a problem including more than one objective in the optimization function. The optimization is carried out as the same time it meets a series of constraints.

In this type of problem, a solution consists on a set of optimal solutions, instead of a single solution. The multi-

objective optimization problem in an algebraic form is the following:

D: Set of object stored in the Cloud associated to a location.

1) Parameters

Let,

Os Object size of each d∈D
Bc Battery consumption by object d∈D
Tc Transfer cost required by each object d∈D
T Time in a point
B Battery level of a mobile node
Sc Storage capacity of a mobile node
Tr Data transfer rate
Cp Cost willing to pay by user

2) Model objectives

- Maximize the set of objects d∈D to be downloaded during a request.

$$Max\sum_i X_i \times Os_i \quad (1)$$

- Minimize power consumption generated by discharge a set of objects d∈D.

$$Min\sum_i X_i \times Bc_i \quad (2)$$

- Minimize transfer costs generated by download the set of objects d∈D.

$$Min\sum_i X_i \times Tc_i \quad (3)$$

3) Constraints

- Discharge capacity constraint.

$$\sum_i X_i \times Os_i \le Sc \quad (4)$$

- Battery consumption constraint.

$$\sum_i X_i \times Bc_i \le B \quad (5)$$

- Transfer cost constraint.

$$\sum_i X_i \times Tc_i \le Cp \quad (6)$$

- Time constraint associated to a location point.

$$\sum_i \frac{X_i \times Os_i}{Tr} \le T \quad (7)$$

The selection of data from the Cloud to be stored in the mobile cache provides as a result a list of objects associated to one F point. These objects conforms a solution to the multi-objective optimization problem, giving acceptable values to the three objective functions, finding a balance between them.

Pocket caching uses the concept of Pareto dominance expressed in the definition 2 to resolve the optimization problem. This method uses different criteria's to compare and decide the better solution.

Definition 1: A solution x dominates, in a Pareto way, a solution$y$, if and only if x is better than y according to at least one optimization objective, and x is not worse than y in any of the other objectives [17].

A solution is Pareto-optimal if there is no other feasible solution that dominates. The set of solutions in a POM, comprises all those Pareto-optimal solutions. The solutions within this set of solutions are called "non-dominated" and conforms all Pareto-optimal frontier [18].

C. POM ś Solution

This proposal uses an a posteriori method to solve the problem. First, it searches solutions and then, performs a decision making process. The first phase uses a meta-heuristic search with an evolutionary algorithm (henceforth AE). The result is a set of Pareto-optimal solutions to the problem. The second phase uses this set and a decision maker to decide a solution according to the preference information.

D. Evolutionary Algorithm for Data Download

An evolutionary approach selects feasible or near feasible solutions in an efficiently and less computationally expensive way. The AEs are based on the species evolution theory of Darwin. The evolution process can be seen as a search to locate individuals that best fit within a changing search space.

In an AE each individual represents a possible solution and the whole set represent the population. Information regarding the individual characteristics in population (genotype) is given in the chromosomes. A chromosome is generally represented as a bit string or data structure. The genotype defines an individual organism, expressed in a phenotype and consists of one or more chromosomes. These chromosomes are composed of separate genes, who take certain values of the genetic alphabet (alleles). The locus identifies the gene position on chromosome [16].

Each possible solution is assigned a fitness value based on how good it is for the problem addressed. This value will be used for the selection process, favoring individuals with better fitness. Then, a subset of these individuals is subjected to a crossover and mutation operator.

Pocket caching uses SPEA2 algorithm [19] to identify the Pareto-optimal frontier. This algorithm is widely used in this type of problem, and evidences good results in terms of performance and convergence of the solution. SPEA2 consists of finding an external set of solutions P', which represents the best Pareto solutions from an initial solution set P. The solution set P evolves in Pareto-optimal solutions that will be included in P' after each iteration.

The basic elements of the evolutionary approach used to solve the optimization problem, is presented in the following.

E. Chromosome Representation

The chromosome represents a solution in the search space of the optimization problem. As illustrated in Fig. 4, the chromosome is a vector of 4-field. The first field (*Cx*) is an objects vector to download, and the other three, *Os*, *Bc* and *Tc* fields are the objective values of the problem, and represent respectively the size of the objects to download, energy consumption and monetary cost generated. *Cx* vector consists of a field denoted as X, which decides whether the object i is or not downloaded.

At a first level, the chromosome corresponds to the values

taken by the solution in each objective function. At a second level, the data object list to be downloaded for each solution is associated. A disadvantage of this type of direct representation is the use of a repairing process during algorithm operations as support to produce feasible solutions. This can affect the algorithm performance.

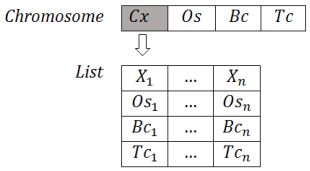

Fig. 4. Chromosome representation.

F. Evolutionary Operators

Individuals exchange information through selection, crossover and mutation operators.

Selection: The selection method used is the binary tournament. A given number of individuals in the population is selected randomly. They are compared with other individuals according to their fitness value similar to a sports competition. The individual with the best value is selected.

Crossover: As illustrated in Fig. 5, the crossover operation consist in select two parent chromosomes and then, a crossing point between 1 and (L/4) is calculated, where L is the length of the chromosome. Parent chromosomes are cut at this point to produce four sub-chains that will be exchanged to generate two offspring chromosomes.

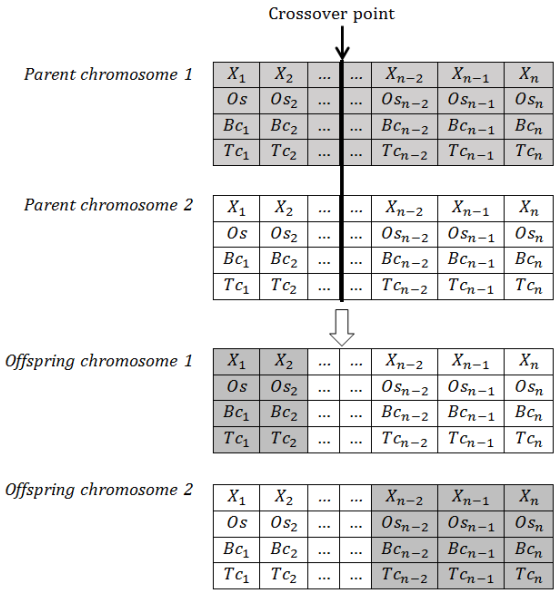

Fig. 5. Crossover operator.

Mutation: The mutation operator, as illustrated in Fig. 6, modifies the value of two bits in the chromosome string.

G. Evolutionary Algorithm Based on SPEA2

This section describes the steps of the algorithm to obtain a list of objects to be downloaded by the mobile node.

1) A P random population with N solutions is created. Additionally, each solution in the population P is validated to define its feasibility, i.e, if they meet all the constraints of the problem.

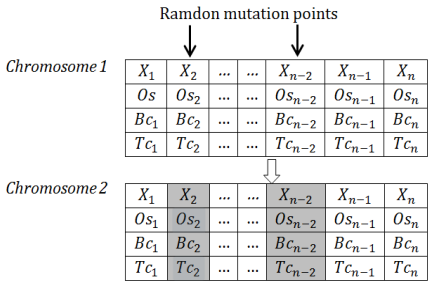

Fig. 6. Mutation operator.

2) For each solution, a level adaptation (fitness function) is calculated. In this case the function takes into account the Pareto dominance concept shown in definition 1. As a result, the dominated and non-dominated solutions sets are identified (Pareto frontier).
3) In SPEA2 the fitness function value of a solution incorporates information of two criteria related with the dominance Pareto concept: The sum of the strength value of the dominating solutions (i.e., the number of solutions that dominates the solution), and the density of the population towards the solution being evaluated. This allows discriminating between individuals with identical fitness values. The calculation is done as discussed in section 3.1 of [19].
4) The values returned by step 2, are used to select the best solutions (with better fitness). A binary tournament method is used on these solutions to select the solutions for the next iteration (i.e., generation). In addition, a truncation operator is used to remove a portion of the elements when the file capacity is exceeded.
5) Operations of crossover and mutation are then applied on the selected best solutions. This allows obtaining a new population to move to the next generation.
6) The process finish using the termination criteria, in this moment, the best solutions set are saved; if not, the process returns to step 2.

VI. System Evaluation

The pocket caching optimization strategy is evaluated using both the algorithm execution performance and the quality of the results. Different test scenarios and results of the evolutionary algorithm assessment, proposed as a solution to the problem of downloading data are used to this evaluation.

A. Test Settings

The proposed scenarios are built based on data described by Pyles in [20] and simulated data. In particular it draws on information on energy consumption that occurs during downloading files from a remote server (e.g., A file download 10Mb of data connected through Wifi 802.11bg network in power saving mode with a 2mbps transfer capacity, consumes about 45joules of battery capacity). Based on this information, an initial dataset used as input for the evolutionary algorithm is generated (see Fig. 7).

Each user must specify the maximum amount willing to pay for one day of application execution. However, the idea is to obtain a monetary savings margin during application execution. This restriction has less influence on the problem that energy restriction.

Data_device.csv

Tp	valTp		
Nb	valNb		
Ca	valCa		
Tr	valTr		
Cp	valCp		
Re	valRe1	...	valReN
To	valTo1	...	valToN
Cb	valCb1	...	valCbN
Ct	valCt1	...	valCtN

Fig. 7. Initial dataset.

B. Quality Result

Six test scenarios were proposed, to validate the quality of the results, simulating a hypothetical situation where a user makes a request for data to the Cloud. First control scenarios, taken as a reference for comparison to the other scenarios arise. During the request from a user, it stays in the same position for about 7 minutes.

For each one of the five scenarios, the cost willing to pay for the user, network transfer capacity, local storage, and battery capacity are modified in a controlled way in order to have elements to evaluate the result Finally, the results for each scenario with respect to the first were analyzed.

Scenario 1 (Common data): As shown in Table I, a mobile node has a Battery level of 20% and a cache reserved storage capacity of 300mb. Currently, the device maintains a 1,99mbps 3G network connection. The user is willing to pay up to $800 per any objects number to be downloaded. The algorithm was executed 10 times and calculated the average values of the objectives to form a final solution.

TABLE I: CONTROL SCENARIO

Time in a point	420s (7min)
Battery level	3867j (20%)
Storage capacity	314572800b (300mb)
Data transfer rate	2085888bps (3G/1,99mbps)
Cost willing to pay	$800

The detail of the selected chromosome as part of the decision making process is presented in the following:

TABLE II: SELECTED SOLUTION CHROMOSOME

Downloaded size	43,8mb de 108,3mb
Energy consumed	445j of 3867j
Monetary cost generated	$688 of $800
Time used	27s of 420s
Object number	17 of 50

Scenario 2 (less energy): The energy available in the mobile node becomes 5% and the other parameters remain the same as in the first scenario.

As expected, the model responds to changes in the input parameters as scenario change. The total size of downloaded objects varies, to respect the limits imposed by the constraints.

TABLE III: LESS ENERGY SCENARIO

Time in a point	420s (7min)
Battery level	900j (5%)
Storage capacity	314572800b (300mb)
Data transfer rate	2085888bps (3G/1,99mbps)
Cost willing to pay	$800

The detail of the selected chromosome as part of the decision making process is presented in Table IV.

TABLE IV: SELECTED CHROMOSOME

Downloaded size	43,3mb de 108,3mb
Energy consumed	440j of 3867j
Monetary cost generated	$683 of $800
Time used	21,8s of 420s
Number of Objects	18 of 50

C. Performance Measurement

Two metrics commonly used to verify the result of the implementation of an evolutionary algorithm were used to validate the algorithm performance: The distance between the Pareto-optimal frontier obtained by the algorithm with respect to the true Pareto-optimal frontier (e.g., Generational distance), and the distribution of the solution obtained in the Pareto-optimal frontier obtained by the algorithm (e.g., Dispersion).

These metrics require an a priori knowledge of the true Pareto-optimal front. To calculate it, an approximation, the whole non-dominated solutions obtained by executing the algorithm with 80 iterations for the scenario 1, was used. The number of solutions obtained in the true Pareto-optimal front was 34.

Generational Distance: The generational distance metric represents how far the obtained Pareto frontier from the true Pareto frontier is and is defined as:

$$GD = \frac{\left(\sum_{i=1}^{n} d_i^{\min^2}\right)^{\frac{1}{2}}}{n} \quad (8)$$

where d^{min^2} is the euclidean distance i (in objective space) between each target vector in the Pareto frontier obtained and its corresponding closest to the true Pareto frontier. A zero result indicates that *PFtrue = PFknown*, while any other value indicates that *PFtrue* deviates from *PFknown*.

Dispersion: Measures the distribution of the Pareto frontier solutions and the distance obtained with the ends of the true Pareto frontier.

$$D = \sqrt{\sum_{i=1}^{q+1}\left(\frac{n_i - \tilde{n}_i}{\sigma_i}\right)^2} \quad (9)$$

In this case, a result close to zero indicates that the Pareto frontier obtained is optimally distributed over the true Pareto frontier.

Results of metrics: The results of the metrics used in this work are presented in Table V.

TABLE V: METRICS RESULTS

Generational Distance	0.05
Dispersion	0.7

The values returned by the metrics show a good algorithm performance, because the closeness between the border frontier obtained, and the expected real border.

VII. CONCLUSIONS AND FUTURE WORK

This paper presents the theoretical proposal Pocket Caching to address a data availability problem in the emerging context of MCC Location-Based applications.

Pocket Caching proposes an innovative way to resolve the problem of selection of data stored in a Cloud to be preloaded in a mobile node using multi-objective optimization. Pocket caching uses SPEA2 algorithm to identify the Pareto-optimal frontier and identify a solution to the problem. The results obtained during the evaluation of the algorithm evidences good performance and quality of algorithm. Some challenges related to user path prediction, object relevance identification, and the use of different mobile nodes to participate in the distributed cache are considered future work. These elements are key factor for the Pocket Caching strategy success.

REFERENCES

[1] Cisco, *Cisco Visual Networking Index: Forecast and Methodology*, pp. 2012-2017, June 2013.

[2] W. Holden, *Juniper Research — Mobile Telecoms Research*, 2013.

[3] O. Sotamaa, "All the world's a botfi ghter stage: Notes on location-based multi-user gaming, " in *Proc. the Computer Games and Digital Cultures Conference*, 2002.

[4] M. Zhu, A. Inge, and O. Rolland. Experiences from the development of the pervasive game cityzombie. [Online]. Available: http://www.idi.ntnu.no/~alfw/papers/cityzombie-paper.pdf

[5] J. L. Shen, S. C. Yan, and X.-S. Hua, "The e-recall environment for cloud based mobile rich media data management," in *Proc. the 2010 ACM Multimedia Workshop on Mobile Cloud Media Computing*, New York, 2010, pp. 31-34.

[6] E. Koukoumidis, D. Lymberopoulos, K. Strauss, J. Liu, and D. Burger, "Pocket cloudlets," in *Proc. International Conference on Architectural Support for Programming Languages and Operating Systems*, Newport Beach, California, March 5-11, 2011, pp. 171-184.

[7] C. Vemulapalli, S. K. Madria, and M. Linderman, "Pre-distributionScheme for data sharing in mobile cloud computing," in *Proc. the First International Workshop on Mobile Cloud Computing & Networking*, Bangalore, India, July 2013, pp. 11-18.

[8] M. Weiser, "The computer for the 21st century," *ACM SIGMOBILE Mobile Computing and Communications Review*, pp. 3-11, New York, July 1999.

[9] R. Fakoor, M. Raj, A. Nazi, M. Di Francesco, and S. Das, "An integrated cloud-based framework for mobile phone sensing," in *Proc. the MCC Workshop on Mobile Cloud Computing*, 1st ed. Helsinki, Finland, August 2012, pp. 47-52.

[10] S. Sen, A. Misra, R. Balan, and L. Lim, "The case for cloud-enabled mobile sensing services," in *Proc. the MCC Workshop on Mobile Cloud Computing*, 1st ed. Helsinki, Finland, August 2012, pp. 53-58.

[11] A. H. Chaparro and C. L. J. Guar ń, "Budgie system: Identifying individuals behavioral-patterns to generate pertinent alerts in a pervasive system using mobile devices," M.S. thesis, Bogot á, Colombia, 2013.

[12] R. Hariharan and K. Toyama, "Project lachesis: Parsing and modeling location histories," in *Proc. 3rd International Conference on Geographic Information Science*, Adelphi, MD, October 2004, pp. 106-124.

[13] L. Liao, D. Fox, and H. Kautz, "Extracting places and activities from gps traces using hierarchical conditional random fields," *International Journal of Robotics Research*, Inc. Thousand Oaks, CA, pp. 119-134, Jan. 2007.

[14] J. H. Kang, W. Welbourne, B. Stewart, and G. Borriello, "Extracting places from traces of locations," in *Proc. the 2nd ACM International Workshop on Wireless Mobile Applications and Services on WLAN Hotspots*, Philadelphia, PA, October 2004, pp. 110-118.

[15] M. B. Kjærgaard, S. Bhattacharya, H. Blunck, and P. Nurmi, "Energy-efficient trajectory tracking for mobile devices," in *Proc. the 9th International Conference on Mobile Systems, Applications, and Services*, Washington DC, June 2011, pp. 307-320.

[16] Y. Donoso and R. Fabregat, *Multi-objective Optimization in Computer Networks Using Metaheuristics*, Boston: Auerbach Publications, 2007.

[17] A. A. Freitas, "A critical review of multi-objective optimization in data mining: a position paper," *ACM SIGKDD Explorations Newsletter* , pp. 77-86, New York, Dec. 2004.

[18] C. A. Coello and G. B. Lamon, *Applications of Multi-objective Evolutionary Algorithms*, Singapore: World Scientific Pub Co Inc, 2004.

[19] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: Improving the Strength Pareto Evolutionary Algorithm," *Computer Engineering and Networks Laboratory (TIK)*, Zurich, June 2001.

[20] A. J. Pyles, X. Qi, G. Zhou, M. Keally, and X. Liu, "SAPSM: Smart adaptive 802.11 PSM for smartphones," in *Proc. the 2012 ACM Conference on Ubiquitous Computing*, Pittsburgh, Pennsylvania, July 2012, pp. 11-20.

Mar á del Pilar Villamil G. is an associate professor at the Department of Systems and Computing Engineering, University of Los Andes, Bogot á Colombia. She did her B.Sc. and M.Sc. degrees in systems and computing engineering at University of los Andes in 1996 and 1998 respectively. She received her PhD degree "Docteur en Informatique" at Institut National Polytechnique de Grenoble, Francia in 2006.

Currently her research fields are in business intelligence, bussiness analytics and data management in large scale contexts.

Carlos Javier Urango M. is a graduate research assistant at the Department of Systems and Computing Engineering, University of Los Andes, Bogot á Colombia. He did his M.Sc. degree in systems and computing engineering at University of Los Andes in 2014. Currently, his research fields are in data mining and mobile and cloud computing.