A Distributed Allocation Strategy for Data Mining Tasks in Mobile Environments

Carmela Comito, Deborah Falcone, Domenico Talia and Paolo Trunfio

Abstract The increasing computing power of mobile devices has opened the way to perform analysis and mining of data in many real-life mobile scenarios, such as body-health monitoring, vehicle control, and wireless security systems. A key aspect to enable data analysis and mining over mobile devices is ensuring energy efficiency, as mobile devices are battery-power operated. We worked in this direction by defining a distributed architecture in which mobile devices cooperate in a peer-to-peer style to perform a data mining process, tackling the problem of energy capacity shortage by distributing the energy consumption among the available devices. Within this framework, we propose an energy-aware (EA) scheduling strategy that assigns data mining tasks over a network of mobile devices optimizing the energy usage. The main design principle of the EA strategy is finding a task allocation that prolongs network lifetime by balancing the energy load among the devices. The EA strategy has been evaluated through discrete-event simulation. The experimental results show that significant energy savings can be achieved by using the EA scheduler in a mobile data mining scenario, compared to classical time-based schedulers.

Carmela Comito
DEIS, University of Calabria, Rende (CS), Italy, e-mail: ccomito@deis.unical.it

Deborah Falcone
DEIS, University of Calabria, Rende (CS), Italy, e-mail: dfalcone@deis.unical.it

Domenico Talia
ICAR-CNR and DEIS, University of Calabria, Rende (CS), Italy, e-mail: talia@deis.unical.it

Paolo Trunfio
DEIS, University of Calabria, Rende (CS), Italy, e-mail: trunfio@deis.unical.it

1 Introduction

A growing number of mobile data intensive applications appeared on the market in recent years. Examples include cell-phone- and PDA-based systems for body-health monitoring, vehicle control, and wireless security systems. Advanced support for data analysis and mining is necessary for such applications. A key aspect that must be addressed to enable effective and reliable data mining over mobile devices is ensuring energy efficiency, as most commercially available mobile devices have battery power which would last only a few hours. Therefore, data mining tasks in mobile environments should be allocated and scheduled so as to minimize the energy consumption of low-capacity mobile devices.

Only very few studies have been devoted on energy characterization of data mining algorithms on mobile devices [1], but not in cooperative distributed scenarios. We worked in this direction by defining a distributed architecture in which mobile devices cooperate in a peer-to-peer style to perform a data mining task, tackling the problem of energy capacity shortage by distributing the energy consumption among the available devices. Efficient resource allocation and energy management is achieved through clustering of mobile devices into local groups, also termed clusters. Such a cooperative architecture can be seen as a set of requestors, i.e., mobile applications generating data mining tasks to be executed, and a clustered set of resources, i.e., mobile devices characterized by different levels of energy and processing power, where tasks can be executed. To make the most of all available resources, a proper distribution of tasks among clusters and individual devices is crucial. The design and evaluation of such energy-aware (EA) task allocation (or task scheduling) strategy is the main goal of this paper.

The design principle of the EA scheduling strategy is to find a task allocation that prolongs network lifetime by balancing the energy load among clusters. To this end, the EA scheduler implements a two-phase heuristic-based algorithm. The algorithm first tries to assign a data mining task locally to the cluster that generated the execution request, by maximizing the cluster residual life. If the task cannot be assigned locally, the second phase of the algorithm is performed by assigning the task to the most suitable node all over the network of clusters, maximizing this way the overall network lifetime. We characterize the energy consumption of mobile devices defining an energy model in which the energy costs of both computation and communication are taken into account.

The EA approach has been introduced in our previous work [2]. In this paper we propose an extensive evaluation of the EA allocation strategy using a custom discrete-event simulator, which allowed us to assess it effectiveness on a large range of data mining tasks. The experimental results show that a significant improvement can be achieved using our EA scheduler compared to the time-based round-robin scheduler. In details, our algorithm: i) is effective in prolonging network lifetime by reducing the energy consumption, without sacrificing the number of tasks completed; ii) in all the experiments performed, it was able to keep alive most of the mobile devices thanks to its energy load balancing strategy.

The remainder of the paper is organized as follows. Section 2 introduces the reference architecture. Section 3 presents the EA allocation scheme. Section 4 presents the experimental results. Section 5 discusses related work. Finally, Section 6 concludes the paper.

2 Reference Architecture

In a mobile ad hoc network, efficient resource allocation, energy management and routing can be achieved through clustering of mobile nodes. In a clustering scheme, mobile nodes are divided into clusters. Generally, geographically adjacent devices are assigned to the same cluster. Under a cluster-based structure, mobile nodes may be assigned different roles, such as *cluster-head* or *cluster member*. A cluster-head normally serves as the local coordinator for its cluster, performing intra-cluster transmission arrangement, data forwarding, and so on. A cluster member is a non cluster-head node without any inter-cluster links.

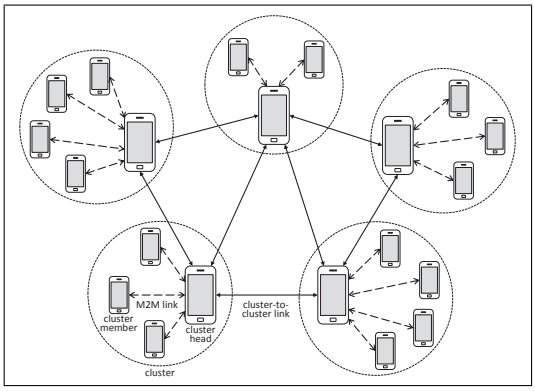

Fig. 1 Reference architecture for mobile-to-mobile collaborations between mobile devices.

In this work we assume, as a reference, the cluster-based architecture shown in Figure 1, which is meant to support mobile-to-mobile (M2M) collaborations between mobile devices. Examples of M2M collaborations occur in several domains such as disaster relief, construction management and healthcare. Mobile nodes within a cluster interact through ad-hoc connections (e.g., wi-fi, bluetooth), that we refer to as M2M links, represented as dashed arrows in the figure. Interactions between clusters (cluster-to-cluster links) take place through ad-hoc connections between the respective cluster-heads, and are represented as continuous arrows in the figure.

The architecture is based on a fully distributed cluster formation algorithm in which nodes take autonomous decisions; no global communication is needed to setup the clusters but only local decisions are taken autonomously by each node.

This means that the proposed architecture is self-organized into mobile clusters: when mobile devices meet each other, i.e., when they are within the same transmission range, they can form a mobile group. The self-organization nature of the clustering scheme distributes the responsibility between all the mobile nodes. We do not focus in this paper on the cluster formation algorithm as it has been presented in a previous work [3].

All types of interactions in the architecture shown in Figure 1 take place either to ask for a computation request, or to perform a distributed allocation of a data mining task, as detailed in the next section.

3 Distributed Task Allocation Strategy

The energy-aware (EA) scheduling strategy deals with a set of independent data mining tasks, dynamically generated over time, which have to be allocated over mobile nodes organized into the cluster-based architecture introduced earlier.

Task allocation is a step of the more general scheduling problem; it can also be seen as a global scheduling or meta-scheduling that distributes the tasks among the devices. Once tasks have been allocated, the problem becomes one of defining a feasible local schedule that manages task execution for each node. In this paper we focus on the task allocation problem and we refer to task allocation or task scheduling interchangeably.

The task allocation problem has been proven to be NP-Complete in its general form [4]. However, some optimal algorithms have been proposed for restricted versions of the problem and some heuristic-based algorithms have been proposed for the more general versions of the problem allowing to find good allocations in polynomial time [5].

We propose a two-phase heuristic-based, decentralized algorithm. When an assignment decision has to be made for a task, the first phase, referred to as *local assignment* phase, is responsible for local task arbitration: it considers the energy consumption of task execution on the different devices within the local cluster. The algorithm tries to minimize the total consumed energy in the cluster by assigning the task to the device that allows to extend the cluster residual life. If the first phase is not feasible, the second phase, referred to as *global assignment* phase, is responsible for task arbitration among clusters: the task will be assigned to the most suitable device, all over the network of clusters, that maximizes the overall network lifetime.

Some definitions and notations are introduced in the following to support the description of the proposed distributed allocation strategy.

- $PC_i(t)$: processing capacity of device d_i at time t.
- $M_i(t)$: memory availability of device d_i at time t.
- $EEC_i(t_j, s)$: estimated energy consumed for computation by device d_i to process a task t_j over a data set of size s.
- $EET_i(t_j, s)$: estimated energy consumed for communication by device d_i to process a task t_j over a data set of size s.

- $EMC_i(t_j, s)$: estimated memory consumption of device d_i to run a task t_j over a data set of size s.
- $EPC_i(t_j, s)$: estimated processing capacity required by device d_i to execute a task t_j over a data set of size s.
- $RL_i(t)$: residual life of node i at time t, defined as follows:

$$RL_i(t) = RE_i(t)/P_i(t) \tag{1}$$

where $RE_i(t)$ is the residual energy available at node i at time t, and $P_i(t)$ the instantaneous power.

The classical task allocation problem can be reformulated here as the problem of finding the proper task assignment that minimizes the energy dissipated in the system. In other words, we formalize the problem of task allocation as an optimization problem. The aim of the optimization is to maximally extend the life of all the nodes in the network by balancing the load proportionally to the energy of each node. We achieve this goal by iteratively trying to improve a candidate solution. A feasible allocation is optimal if the corresponding group residual life (in case of local assignment) or system lifetime (in case of global assignment) is maximized among all the feasible allocations.

The candidate nodes to which a task t_a could be assigned have to satisfy the following constraints:

1. a node d_i must have enough processing power to perform the task over a data set of size s: $EPC_i(t_a, s) < PC_i(t)$
2. a node d_i must have enough energy to perform the task over a data set of size s: $EEC_i(t_a, s) < RE_i(t)$
3. a node d_i must have enough memory to perform the task over a data set of size s: $EMC_i(t_a, s) < M_i(t)$

During the local assignment phase, a cluster-head, or the set of neighboring cluster-heads in case of the global assignment, will choose the local node, among the ones satisfying the above constraints, that will prolong the life of the corresponding local group by using the following objective function:

$$RL_{LG_j}(t) = \text{Max} \sum_{i=1}^{N_{LG_j}} \alpha_i RL_i(t) \tag{2}$$

where RL_{LG_j} denotes the residual life of local group LG_j, N_{LG_j} is the number of nodes within the local group LG_j, RL_i is the residual life of node i in the group, and parameter α_i takes into account the importance of node i in the local group. The node associated with the maximum value in the objective function will be selected by the cluster-head as candidate node. Note that throughout the experimental evaluation presented in the next section, the parameter α_i is set to 1 thus, all the nodes have the same role within the local group.

If the global assignment phase is activated, the final decision is taken by considering all the candidate nodes proposed by the neighboring clusters. The task will be assigned to the local group that maximizes the life of the whole network:

$$RL_{net}(t) = \text{Max} \sum_{j=1}^{N} \alpha_j RL_{LG_j}(t) \quad (3)$$

where N is the number of groups in the network.

4 Experimental Results

In this section we present an experimental evaluation of the proposed EA scheduler, performed using a custom discrete-event simulator. As a first step, the simulator builds a network composed of 100 mobile devices, and let them grouping into clusters based on the algorithm described in [3]. Then, an initial energy capacity ranging from 3,000 J to 11,000 J is assigned to each device, following a normal distribution. After the initial setup, mobile devices start generating a set of data mining tasks to be executed, which are allocated to the available nodes according to the EA strategy described in the previous section.

To the purpose of our simulation, we characterize the data mining tasks on the basis of the energy required to complete their execution. To this end, we selected three reference data mining algorithms (J48, for data classification; Kmeans, for data clustering; and Apriori, for association rules discovery). Each algorithm was used to analyze a sample dataset (of varying size), using an Android smartphone as mobile device. After each execution we measured the actual energy consumed to perform the task, which was used as input for the simulations.

The simulation aims at studying the behavior of the scheduler with respect to the energy depletion and network lifetime. Accordingly, as performance metrics, we use the *number of alive devices*, the *number of completed tasks*, and the *network residual life* at the end of the simulation. To assess the effectiveness of the EA strategy, we compared its performance with the one achieved by round-robin (RR) scheduling algorithm.

In a first set of experiments, for each reference algorithm (J48, Kmeans, or Apriori), we ran a set of tasks by varying the size of the dataset to be mined from 100 kB to 3.2 MB. Each simulation lasts 30 hours, with tasks that arrive following a Poisson distribution with a frequency $\lambda = 160$ tasks per hour. Figures 2(a), 3(a) and 4(a) show the number of alive devices at the end of the simulation for J48, Kmeans, and Apriori, respectively, using the EA and RR strategies. With all data mining algorithms and dataset sizes, the number of alive devices with EA is greater than (and in a few cases equal to) that of RR. In particular, Figures 2(a) and 3(a) show that there are no alive devices with RR for datasets greater than 200-400 kB using J48 and Kmeans. In contrast, in the same configurations, EA keeps alive a high percentage of the devices. Additionally, we can note that, with EA, the number of alive nodes increases with the dataset size. This is due to the fact that, when the dataset size increases, also the energy required to complete the task increases. Since the EA scheduler does not allocate tasks when the available devices do not have enough energy, this lead to a higher number of alive nodes. It is important to note that the

higher number of alive devices ensured by EA compared to RR, is obtained without reducing the number of tasks completed, as shown in Figures 2(b), 3(b) and 4(b). Figure 5(a) shows the network residual life measured at the end of the experiments. The figure confirms that the EA scheduler is effective in prolonging network lifetime compared to the RR algorithm.

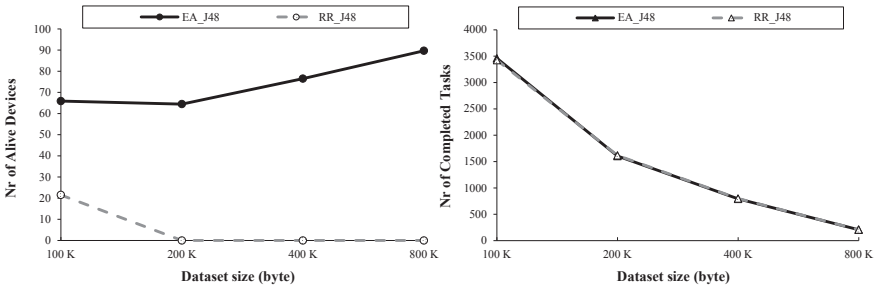

Fig. 2 (a) Number of alive devices and (b) Number of completed tasks w.r.t. dataset size, using EA and RR with J48.

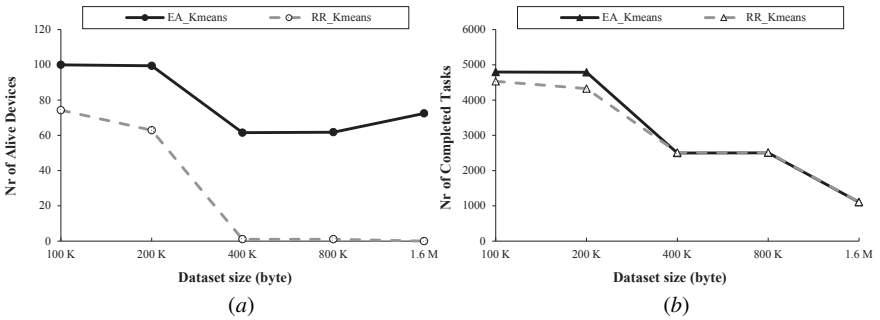

Fig. 3 (a) Number of alive devices and (b) Number of completed tasks w.r.t. dataset size, using EA and RR with Kmeans.

In a second set of experiments, for each reference algorithm (J48, Kmeans, or Apriori), we ran a set of tasks with a fixed dataset size (200 kB) but with task arrival rate λ varying from 80 to 1280 tasks per hour. Figures 5(b) shows the network residual life measured at the end of the experiments for the three algorithms, using EA and RR. As expected, increasing the task arrival rate, the network residual live tends to zero both for EA and RR. However, for the lightest of the three data mining algorithms (Apriori), the residual life does not reach zero and the difference between EA and RR increases with λ in favor of EA. Figure 6(a) shows the number of alive devices for the three algorithms, using EA and RR. The results demonstrate, also in this case, that the number of alive devices with EA is greater than that

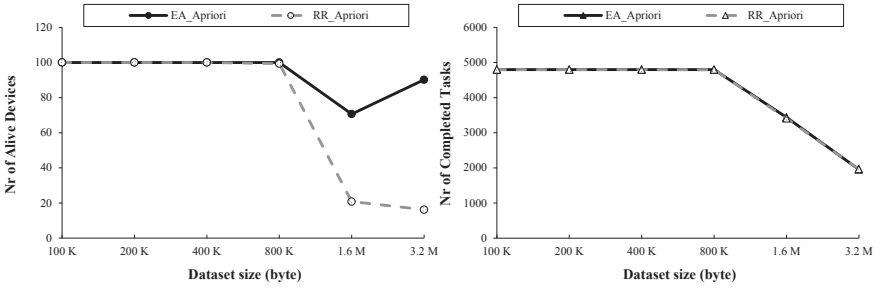

Fig. 4 (a) Number of alive devices and (b) Number of completed tasks w.r.t. dataset size, using EA and RR with Apriori.

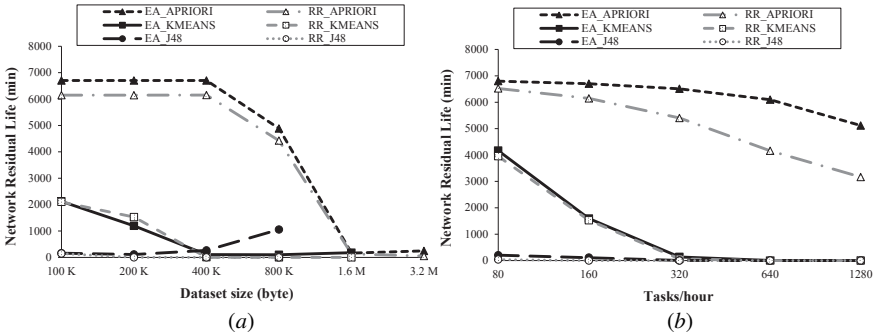

Fig. 5 Network residual life using EA and RR with Apriori, Kmeans and J48, w.r.t. (a) dataset size; (b) task arrival frequency.

achieved by RR. Figure 6(b) compares the performance of EA and RR in terms of completed tasks for the three algorithms. With Apriori, both EA and RR are able to complete more tasks as λ increases, but EA ensures better performance. With J48 and Kmeans, over a given task arrival rate, the number of completed tasks cannot increase because the network residual life is zero, as shown in Figures 5(b). However, even in this cases, there is a slight advantage for EA compared to RR for $\lambda < 320$ tasks per hour.

The experimental results discussed above demonstrated that a significant improvement can be achieved using the EA scheduler compared to the RR scheduler in a distributed mobile scenario. In details, our algorithm: i) resulted effective in prolonging network lifetime by reducing the energy consumption, without sacrificing the number of data mining tasks completed; ii) in all the experiments performed, it was able to keep alive most of the mobile devices thanks to its energy load balancing strategy.

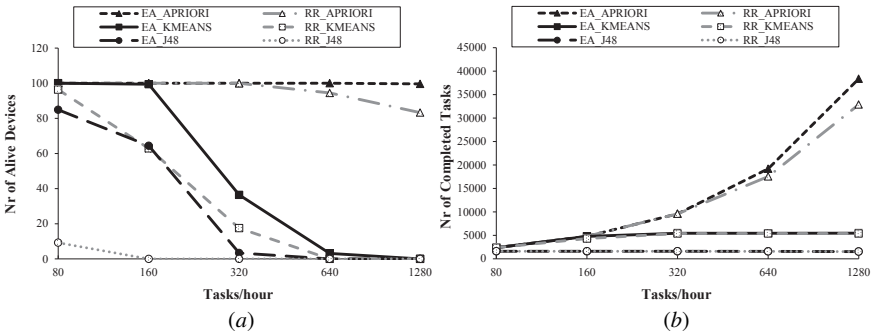

Fig. 6 (a) Number of alive devices and (b) Number of completed tasks w.r.t. task arrival frequency using EA and RR with Apriori, Kmeans and J48.

5 Related Work

Most of the existing research work in the area of energy-aware systems are hardware-based techniques focusing on reducing the energy consumption of the processor. One of the most adopted techniques is turning off idle components [6]. Dynamic Voltage Scaling (DVS) is another technique of energy conservation. DVS refers to the technique of simultaneously varying the processor voltage and frequency as per the energy performance level required by the tasks [7, 8, 9]. Remote execution is a software-based technique in which a device with limited energy transfers a computational task to a nearby device which is more energy powerful. Energy-aware task scheduling is another software method where the scheduling policy aims at optimizing the energy.

To the best of our knowledge, little work has been done on energy-aware scheduling over a mobile ad hoc networks (MANETs), and not for data mining scenarios. In [10] an energy-aware dynamic task allocation algorithm over MANETs is proposed. However, this work is different from ours, both for the considered application scenarios and for the underlying architecture and cost function to be optimized. We group the devices in clusters to promote local cooperation among nearby devices and to minimize the transmission energy. This issue is particularly relevant because we have experimentally found that the transmission energy highly impacts on the overall energy consumption. In contrast to ours, the solution proposed in [10] is effective for compute intensive applications and does not address the communication aspects of the system. Furthermore, we adopt a different objective function: we maximize the network residual life rather than minimizing the energy consumption.

Using the residual life parameter we are able to actually consider the real energy consumption rate of single devices, single clusters and the overall network. Conversely, [10] does consider only the local computation issues and it works at a node level ignoring the workload in the rest of the network. Thus, differently from us, they do not take into account the actual load of the devices with the possibility of assigning a task to a device that consumes less energy, but which is less charged

compared to another one that consumes more energy but it is more energy powerful and thus could efficiently execute that task.

6 Conclusions

Supporting data mining in mobile environments requires effective architectures and allocation strategies to improve energy utilization of battery-operated devices. We addressed this issue by defining a distributed architecture in which mobile devices cooperate in a peer-to-peer style to perform a data mining process, tackling the problem of energy capacity shortage by distributing the energy consumption among the available devices.

Within this framework, we proposed an energy-aware (EA) task allocation scheme focusing on energy efficiency. To conservatively consume energy and maximize network lifetime the EA adopts a heuristic algorithm that balances the energy load among all the devices in the network. Experimental results show that significant improvements in terms of residual network lifetime, number of alive devices can be achieved by using the EA scheduler in a mobile data mining scenario, compared to classical time-based schedulers such as round-robin.

Acknowledgements This work is partially funded by European Commission, European Social Fund (ESF), and Regione Calabria.

References

1. R. Bhargava, H. Kargupta, M. Powers. Energy Consumption in Data Analysis for On-Board and Distributed Applications. *ICML'03 Workshop on Machine Learning Technologies for Autonomous Space Applications* (2003).
2. C. Comito, D. Falcone, D. Talia, P. Trunfio. Energy Efficient Task Allocation over Mobile Networks. *IEEE CGC'11*, 380-387 (2011).
3. C. Comito, D. Talia, P. Trunfio. An Energy-Aware Clustering Scheme for Mobile Applications. *IEEE Scalcom'11*, 15-22 (2011).
4. R. Garey, D. Johnson. Complexity Bounds for Multiprocessor Scheduling with Resource Constraints. *SIAM J. Computing*, 4:187-200 (1975).
5. H. W. D. Chang, W. J. B. Oldham. Dynamic Task Allocation Models for Large Distributed Computing Systems. *IEEE Trans. Parallel Distrib. Syst.*, 6:1301-1315 (1995).
6. K. Li, R. Kumpf, P. Horton, T. Anderson. A Quantitative Analysis of Disk Driver Power Management in Portable Computers. *Winter 1994 USENIX Conference*, 279-292 (1994).
7. J. Zhuo, C. Chakrabarti. An Efficient Dynamic Task Scheduling Algorithm for Battery Powered DVS Systems. *ASP-DAC'05*, 846-849 (2005).
8. Y. Zhang, X. Hu, D. Chen. Task Scheduling and Voltage Selection for Energy Minimization. *DAC'02*, 183-188 (2002).
9. H. Aydin, R. Melhem, D. Moss, P. Mejia-Alvarez. Power-Aware Scheduling for Periodic Real-Time Tasks. *IEEE Trans. Computers*, 53(5):584-600 (2004).
10. W. Alsalih, S. G. Akl, H. S. Hassanein. Energy-Aware Task Scheduling: Towards Enabling Mobile Computing over MANETs. *IPDPS'05*, 242a (2005).