Energy-Aware Heuristics for Mobile Agents Itinerary with Time Constraints

Luciana Rech, Carlos Montez and Rômulo de Oliveira
Department of Automation and Systems Engineering,
Programa de Pós Graduação em Engenharia Elétrica – UFSC - SC, Brazil,
{lrech, montez, romulo@das.ufsc.br}

Abstract. *In some mobile agent applications there may be the necessity of meeting a firm end-to-end deadline, although with some itinerary flexibility. The objective of this paper is to propose heuristics that deal with the cost of physically moving agents; and to evaluate and compare the performance of these heuristics with others for routing real-time imprecise mobile agents in a distributed environment.*

1 Introduction

Mobile agents are small decision-making programs capable of migrating autonomously from node to node in a computer network. They are an effective way to reduce energy consumption, network load and latency [1]. They execute in asynchronous and autonomous way, and can dynamically adapt themselves, thus establishing a new paradigm for the programming in distributed environments.

This paper deals with code mobility in a distributed system associated with timing constraints. We assume that the agent’s itinerary possibilities are ignored before the mobile agent's departure, and we will compare the performance of some heuristics according to different quality metrics. The work focuses the energy cost that the mobile agent spent traveling by the network. We describe and analyze new heuristics (along with the heuristics previously presented in [2]) for the dynamic determination of their itinerary following a specific resources diagram (mission).

One possible application scenario is a factory where the agent has the function of gathering data in order to form an image of a problem and to decide the next visit. In this case, the mobile agent reaches the fault diagnosis by traveling through several nodes and by collecting data for decision-making.

Although real-time constraints and code mobility are important requirements, there are few works that deal with both areas. In [3] a routing algorithm based on mobile agents is proposed. Mobile agents have been proposed for efficient data dissemination in ad-hoc networks and wireless sensor networks [4, 5, 6, 7]. In [4] it is proposed a routing approach to forward messages across network partitions. This approach attempts to minimize deadline misses and the energy consumption by scheduling work tasks. In [5] a mobile agent is used to reduce information redundancy and communication overhead in wireless sensor networks. In [7] a mobile agent-based paradigm for data fusion in distributed sensor networks was presented. It proposed an event-driven adaptive method to implement a routing strategy based on genetic algorithm.

2 Overview of the Execution Model

The execution model considered in this paper was previously defined in [2, 8]. Each mobile agent has a mission. The agent's mission is formed by a group of resources to be executed while meeting a deadline. The mission is associated with the visit to a certain group of nodes of the system. These nodes host the necessary resources for the agent to accomplish its mission. In this paper it is assumed that this agent does not communicate with any other agent. There is a benefit associated with each resource based on its functional importance within a particular mission. There is a set of resources in each node of the system.

In each node N_i of the system there is a set R_i of one or more types of resources belonging to set R. Each resource type belonging to set R can be found in one or more nodes, that is, several instances can exist (replicas) of the same resource type, but in different nodes.

An itinerary I is defined as a route traveled in the distributed system. In this execution model the mobile agent has some level of flexibility for the definition of its itinerary. This flexibility is related to some resources that can: (i) be used in any order, (ii) be replicated, (iii) be discarded, or (iv) be partially used.

The definition of the itinerary should take into account: (i) the time of computation for maximum benefit of each task x_i, called C_i; (ii) the communication latency between nodes N_k and N_y, called L_{ky}; (iii) the dependences between the resources that appear in the resource diagram of the mission.

The agent starts at an external node $N0$ (node origin) and it must return to this same node at the end of the mission. Clock synchronization is assumed in the system; and every mission M has a firm deadline D. The mobile agent Z should complete the mission (by returning to node $N0$) before D, otherwise it loses all the benefit collected along the itinerary. The cost for an agent to arrive at a node and to acquire the resource must be considered along with the computation time C_i. For each $r_i \in R$, the benefit $B_{i,t}$ obtained by using resource r_i for the time interval t is given by:

1)** r_i **of type variable $\qquad B_{i,t} = B_i \times \min(1, t/Ci)\,.$ $\qquad$ **(1)**

where B_i is the maximum benefit obtained from r_i and C_i is the maximum computation time associated with r_i (C_i is the time that the agent must stay with the resource in order to collect B_i and t is the time interval for which it actually used the resource).

2)** r_i **of type normal $\qquad$ **(2)**

$$B_{i,t} = B_i \quad \text{if } \; t \geq C_i\,.$$
$$B_{i,t} = 0 \quad \text{if } \; t < C_i\,.$$

In the above equations the precedence and optionally relations must be considered: (i) if r_i precedes r_j, and r_i was not utilized, then $B_j=0$; ii) if r_i’ and r_i” are replicas of r_i, and r_i’ was already utilized, the maximum benefit from r_i” will be zero.

The effective benefit of the mission is obtained by the sum of the benefits collected from each resource in the itinerary:

$$B = \sum B_i(y) \quad \textbf{for each} \quad r_i \in R \qquad \textbf{(3)}$$

where $B_i(y)$ is the benefit realized by the utilization of resource r_i that the agent visited in order to fulfill its mission through itinerary y, B is the benefit generated by the utilization of all resources in the mission.

Classic optimization approaches cannot solve the problem described in this paper, because the resource diagram is not completely known at the starting node. Also, the visited nodes during the itinerary have limited processing capacity.

3 Mobile Agents and Energy-Constraint Scenarios

Differently from [2, 8], in this paper we consider the energy costs of the mobile agents during their mission. The energy consumed by the mobile agent during its mission depends on which nodes it visits. Specific values are assumed to represent the cost associated with the agent’s movement between nodes. In specific scenarios (e.g.

wireless sensor networks) communication consumes much more energy than sensing and computation. In our model, the energy spent is associated only with the communication and not with the processing. The energy cost of the mission is obtained by the sum of the energy costs collected from each node visited in the itinerary:

$$E = \sum E_{Ni\_Ny}(y) \qquad \textbf{for each} \quad Ni, Nj \in \mathrm{N} \qquad (4)$$

where $E_{Ni\_Ny}(y)$ is the energy cost by the agent travel from node N_i until node N_j, that the agent visited in order to fulfill its mission through itinerary y. E is the energy spent by the visited to all nodes in the mission. Simulations will be made in function of the following scenarios:

3.1 Energy-limited scenario

In this scenario, the mobile agent has a limited quantity of energy to accomplish its mission. When agent notices that the energy is over, it returns to origin node ***N0*** and it gets the current benefit. The response time R of the mobile agent mission must met a deadline D, and the agent should do it using the available energy E_{max}. When mobile agent spent all available energy, it comes back to the origin node with the feasible ***max B***, that is, the B that the mobile agent got until current time: $\{ R \leq D ;\ E \leq E_{max} ;\ max\ B\}$.

3.2 Minimum benefit scenario

In this scenario the most important requisite is to obtain the necessary benefit. The response time R of the mobile agent mission must met a deadline D, and the agent should do it using the minimal possible energy ***min E*** and get a necessary benefit ***B.*** When mobile agent gets B, it can go back to the origin node: $\{R \leq D;\ B \geq B_{min} ;\ min\ E\}$.

4 Algorithms for the Definition of the Itinerary

This section presents some heuristics for the mobile agent itinerary definition. Some of them were originally defined in [2]. The heuristics used are simple (they require minimum computational effort) and myopic (they work based on a partially known resource diagram). Whenever the agent needs to decide between two routes that are equivalent (in the perspective of the heuristic) a random decision is made.

These heuristics make their decisions based only on the possible next nodes to be visited. They do not consider the future unfolding of their short-term decisions. The node conditions (system load) and the network conditions should be considered. The agent's decision making should take into consideration the benefit obtained by the execution of a certain resource together with the necessary computation time and the spent energy.

4.1. Lazy Algorithm

It tries to execute the resources in the fastest possible way, ignoring their benefits. Optional resources are not executed, and those with stereotype <<*variable*>> are executed during the smallest possible time. Whenever there is an alternative (when there is not a precedence relation between two resources), it will be chosen the resource that presents the smallest computation time.

4.2. Greedy Algorithm

Whenever there is an alternative route, it will be chosen the resource that presents the largest benefit for the mission. This algorithm always executes optional nodes. In cases where the resource is of the type <<*variable*>>, the whole time requested to obtain the maximum benefit will be executed. When there are replicas of a resource, preference will be given to the resource located in the closest node.

4.3. Random Algorithm

This algorithm chooses randomly the next node to be visited by the agent. This behavior is only possible if there are several instances of a certain resource or there is not a precedence relation between the next resource and some of the other resources yet to be executed. In cases where the resource is of the type <<*variable*>>, it will be randomly chosen a value between ***0*** and the necessary time of computation to reach the maximum benefit from the resource.

4.4. Higher Value Density Algorithm

This algorithm is based on AVDT (Average Density Threshold) [9]. It chooses to be the next resource that one that presents the best benefit/execution time rate. This characteristic is possible only when there is not a precedence relation between two resources. The density of the benefit of the resource is defined by the equation:

$$db_i = B_i / (C_i + L_{j,i} + Q_i) . \qquad \textbf{(5)}$$

where: $\boldsymbol{db_i}$ = benefit density of resource *i*, $\boldsymbol{B_i}$ = benefit obtained by executing resource *i*, $\boldsymbol{C_i}$ computation time used by resource *i*, $\boldsymbol{L_{j,i}}$ is the time necessary to move the agent to node *Ni*, and $\boldsymbol{Q_i}$ is the time spent in the local processor queue. Estimated values can be used when the exact values are not known. The agent maintains the average value until the present moment. When a resource is optional, it will be executed only if its density is higher than the average density of the mission to that moment. In cases where the resource is of the type <<*variable*>>, they are executed randomly.

4.5. Higher Energy Density Algorithm

The Higher Energy Density Algorithm chooses the next resource that one that presents the best benefit/energy_cost rate. This characteristic is possible only when there is not a precedence relation between two resources. The density of the benefit of the resource is defined by the equation: $db_i = B_i / E_i$. **(6)**

where: $\boldsymbol{db_i}$ = benefit density of resource *i*, $\boldsymbol{B_i}$ = benefit obtained by executing resource *i*, and $\boldsymbol{E_i}$ = the energy spent to arrived in a node that hosted the desired resource.

Estimated values can be used when the exact values are not known. The agent maintains the average value until the present moment. When a resource is optional, it will be executed only if its density is higher than the average density of the mission to that moment. In cases where the resource is of the type <<*variable*>>, they are executed randomly.

4.6. Low Energy Algorithm

The decision making of this algorithm is based on the smallest energy cost. The choice about the next resource that must be executed in a mission (only if there is not precedence relation between two resources) is made based on the energy spent to get this resource. When there are replicas of a resource, the replica hosted in a node that presents the smaller travel cost from the real node is chosen. It tries to execute the resources in the more economic way, ignoring its benefits. Optional resources are not executed, and those with stereotype <<*variable*>> are executed randomly.

5. Simulation Conditions

The heuristics were simulated considering the node diagram of Figure 1b. After the execution of each resource, some benefit is added to the total benefit of the mission. The objective of the mission is to reach the greatest benefit while respecting the specified deadline. We assume a system with 9 nodes. The available resources at each node are: Node 1 has (r1,r3); Node 2 has (r8, r12); Node 3 has (r7,r9); Node 4 has (r5,r6); Node 5 has (r4,r6,r12); Node 6 has (r3,r6); Node 7 has (r8,r11); Node 8 has (r2,r13).

For the calculation of the total computation time of the mission it was considered:

- The computation time of each resource (c_1=5, c_2= 7, c_3= 8, c_4=12, c_5=10, c_6=5, c_7=15, c_8=8, c_9=4, c_{10}=[0,10], c_{11}=8, c_{12}=7, c_{13}=2);
- The processor queue time, according to an uniform distribution between 1 and 3 (light load) or between 1 and 10 (heavy load);
- Communication time ***L*** between nodes (exponential distribution with average 2).

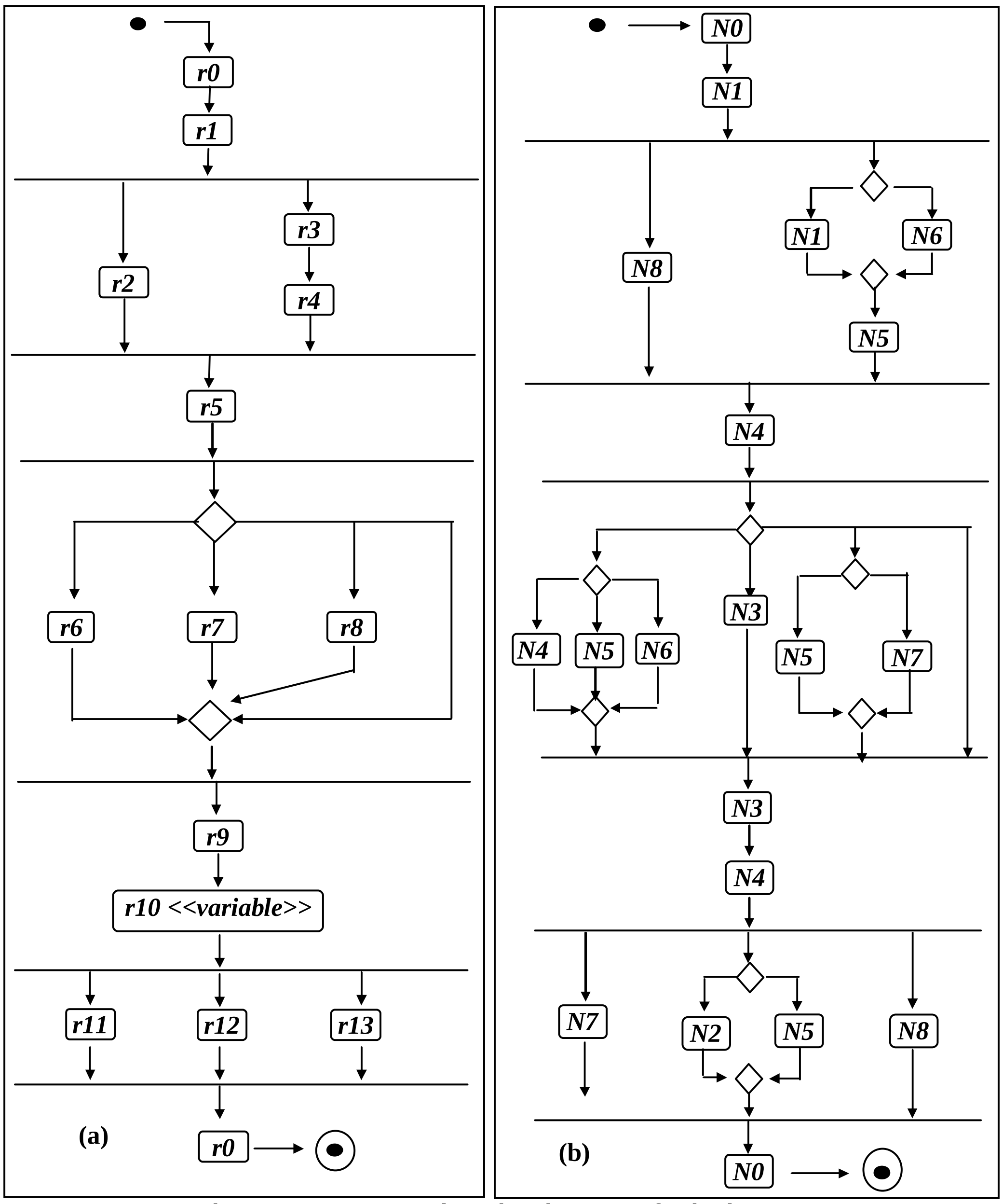

Fig. 1. Resource and Node Diagram of mission *M*.

The quality of the algorithm (global benefit ***G*** of the algorithm) is calculated by the following equation: $G = \sum B\_DA \ / \ NT$ **(8)**
where: ***B_DA*** is the benefit obtained by the utilization of resources while meeting the deadline (B_1=7, B_2=10, B_3=3, B_4=8, B_5=15, B_6=7, B_7=10, B_8=11, B_9=2, B_{10}=[0,10], B_{11}=3, B_{12}=15, B_{13}=3); and ***NT*** is the number of attempts (met + missed deadlines).

The energy cost among the nodes is given by: `N0_N1=4; N1_N5=1; N1_N6=5; N1_N8=4; N2_N0=4; N2_N3=2; N2_N7=5; N2_N8=5; N3_N4=5; N4_N2=5; N4_N3=5; N4_N5=4; N4_N6=5; N4_N7=1; N4_N8=1; N5_N0=4; N5_N3=5; N5_N4=4; N5_N7=4; N5_N8=4; N6_N3=2; N6_N5=5; N6_N8=5; N7_N0=4; N7_N2=5; N7_N3=5; N7_N5=4; N7_N0=1; N8_N0=4; N8_N1=4; N8_N2=5; N8_N4=1; N8_N5=4; N8_N6=5; N8_N7=1.`

The resource diagram of a mission (Figure 1) describes the precedence relations among resources. Any resource used in disagreement with this diagram brings no benefit for the mission. In this diagram the flexibility and the options of the agent accomplish the mission are described. The resource diagram is gradually known by the mobile agent along the route, because it is defined by the information that the agent accesses at each node. Resource type ***r0*** is a pseudo-resource, it is found only in node ***N0*** and it indicates that the agent must return to the original node. The node diagram presents the possibility of many different itineraries that the agent may take in order to execute its mission (Figure 1b). In this diagram, each resource (showed in the resource diagram) is replaced by the node or nodes where that resource appears in the system.

5.1. Simulations Results

Figure 2 presents the behavior of the algorithms for situations where the system load is light (graphs of the left column) and heavy (right column). In (a) and (f) it is shown the global benefit ***G*** obtained by each algorithm, in (b) and (g) it is shown the energy ***E*** that is spent with the agents transferences, in (c) and (h) it is shown the rate of agents that met their respective deadline, in (d), (e), (i) and (j) it is shown the global benefit ***G*** obtained with relation to the spent energy ***E*** for specific deadlines. Figure 2 presents two different legends, the left legend represents the graphs (a), (b), (c), (f), (g), (h); and the right legend represents the graphs (d), (e), (i), (j).

Analyzing the graphic 2(a) it can be noticed that from a certain moment (point in that all the agents meet their mission deadlines) the global benefit obtained by the algorithms stays stable, independently of the remaining time before the deadline. With loose deadlines there is a maximum threshold to the global benefit ***G***. The heuristic that get the biggest ***G*** with loose deadline is the *Greedy* algorithm. *Low Energy* and *Energy Density* algorithms get a ***G*** smaller than the Greedy algorithm, because their decision making is based on the energy consumption and not on the resource benefit. With respect to the obtained benefit, the algorithms *Lazy* and *Greedy* represent the two extremes. Algorithms *Value Density*, *Energy Density* and *Low Energy* presented close and not expressive results when compared to the performance of the other heuristics.

The *Low Energy* and *Energy Density* algorithms obtained similar results, because both consider only the energy spent ***E*** (others algorithms consider ***B*** or ***C*** to decide which the next resource must be executed). These heuristics consider the energy that the agents will spend traveling among the nodes. Whenever the agent choose to stay in the same node, the energy is ***E=0***, in this case the *Low Energy* algorithm always will choose this itinerary because there isn't ***B*** or ***C*** values that compensate the energy economy.

Considering only the energy consumption aspect (Figures 2(b) and 2(g)), the best result was obtained by the *Low Energy* algorithm. This algorithm presented the smallest energy spent with benefit enough to complete the mission successfully, respecting the mission deadline. Each heuristic obtained the same energy consumption for light and heavy loads. The Greedy algorithm spent more energy than other heuristics.

With respect to the rate of agents that met their deadline (Figure 2(c) and 2(h)), the *Lazy* algorithm was the first to meet all deadlines. In the perspective of this metric, the *Lazy* agent will always get the best results with any deadline.

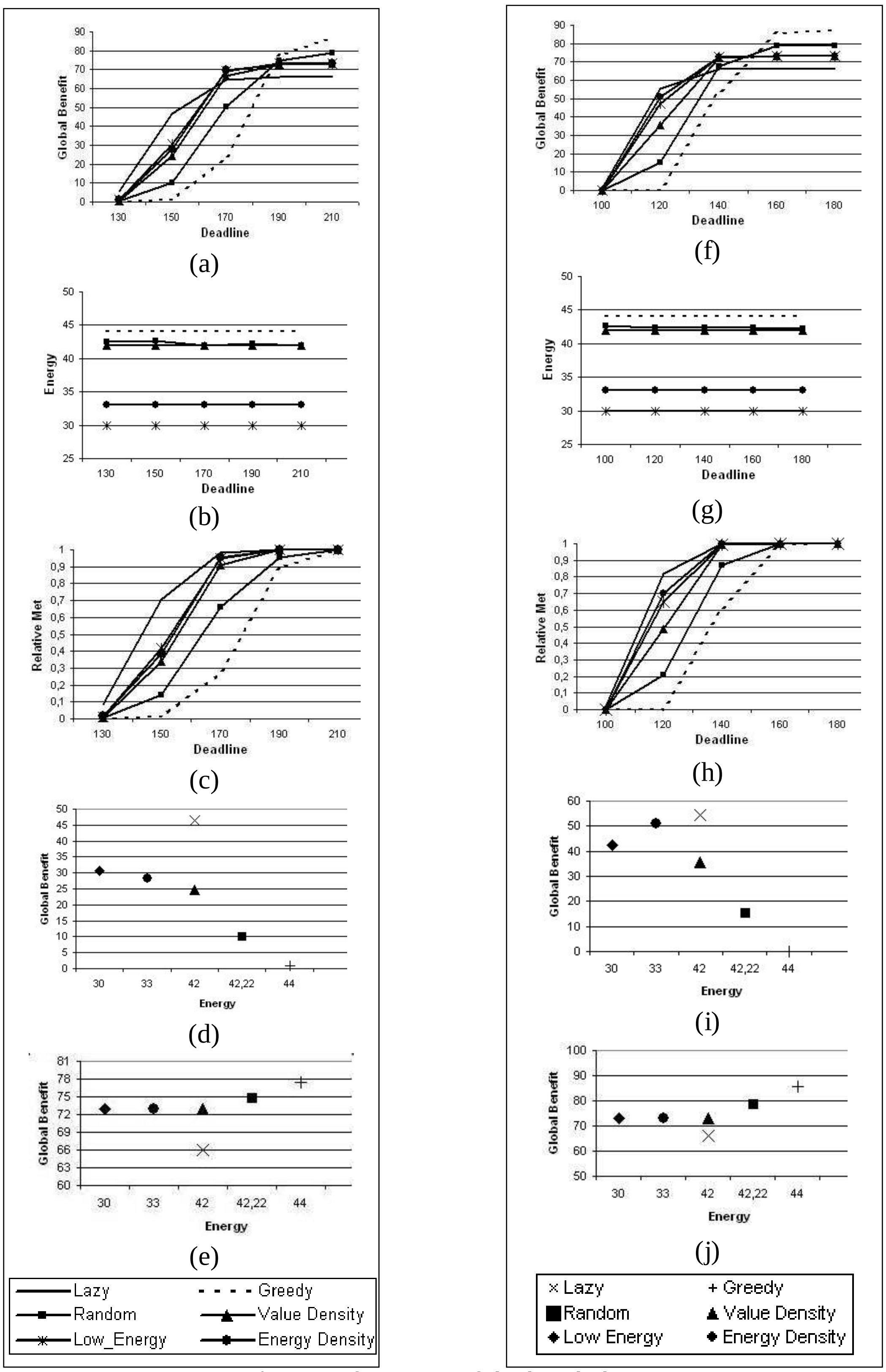

Fig. 2. **Performance of the heuristics.**

Each heuristic has fixed consumption energy (with exception of *Random* algorithm). This value is characterized by the decision-making factor that the heuristics uses to define the next resource to be executed. The heuristic spent the same energy with tight or loose deadlines, that is confirmed for the graphs 2(d) with ***D= 120***, 2(e) with ***D=160***, 2(i) with ***D=150*** e 3(j) with ***D=190*** where are presented the relations between the global benefit and the consumption energy for different values of deadline and different system situations (light and heavy load).

Figures 2(f) to (j) presents the behavior of the algorithms when a heavy system load is simulated. The results are similar to those of a lightly loaded system. It is important to notice that, in this case, the total time to finish the mission is greater because the time that the agent waits in the local processor queue before using the resource in each node.

The decision about the best heuristic will always depend on the deadline being loose or tight. This decision also depends on the purposes and objectives of the mission. For example, if there is an energy-limited environment, it is recommended *Low Energy* or *Energy Density* algorithms to the itinerary definition. But if the objective is to get maximum benefit and the deadline is loose, the best choice is the *Greedy* algorithm. However if there is limited time, the *Lazy* algorithm must be the better choice.

6. Conclusions

Nowadays, energy efficient routing algorithms have emerged as one of the key growth areas for distributed systems. Researches have been mostly concerned in propose approaches trading-off energy with requirements like time constraints, benefit and others. This paper presented new heuristics that consider energy consumption, along with other heuristics for routing real-time imprecise mobile agents in a distributed environment. The agents have firm deadlines but some flexibility for the definition of their itinerary. There are time constraints and the concept of optional resource and precedence added to the technology of mobile agents.

Low Energy Algorithm obtained good results relating to energy save, followed for the Energy Density Algorithm. The decision about the best heuristic will always depends on how loose or tight is the deadline. This decision also depends on the purposes and objectives of the mission.

References

[1] Lange, D. and Oshima, M. Seven Good Reasons for Mobile Agents. Communication of the ACM. Vol.42 1999 (88-89)

[2] L. Rech, R. de Oliveira, and C. Montez. Dynamic Determination of the Itinerary of Mobile Agents with Timing Constraints. IAT 2005 IEEE/WIC/ACM Int. Conf. on Intelligent Agent Technology. Compiègne. France. Sep. (2005) 45-50.

[3] W. Qu, H. Shen, and Y. Jin. Theoretical Analysis on a Traffic-Based Routing Algorithm of Mobile Agents. IAT 2005 IEEE/WIC/ACM Int. Conf. on Intelligent Agent Technology. Compiègne. France. Sep. 2005 (520-526).

[4] J. Brustonoli, S. Khattab, D. Mossé, et al. Integration of Application-Layer Scheduling and Routing in Delay-Tolerant MANETs. Pitt CSD T.R. 2006

[5] M. Chen, T. Kwon, and Y. Choi. Data Dissemination based on Mobile Agent in Wireless Sensor Networks. Proceedings of the IEEE Conference on Local Computer Networks (LCN 05), IEEE Computer Society. 2005

[6] H. Qi, Y. Xu, and X. Wang. Mobile-Agent-Based Collaborative Signal and Information Processing on Sensor Networks. Proc. of the IEEE. Aug. 2003. Vol.91. N°8. (1172-1183)

[7] Q. Wu, N.S.V. Rao, J. Barhen, et al. On computing mobile agent routes for data fusion in distributed sensor networks. IEEE Transactions on Knowledge and Data Engineering. Vol. 16. N° 6. June. 2004 (740-753)

[8] L. Rech, R. de Oliveira, and C. Montez. A Clone-Pair for the Dynamic Determination of the Itinerary of Imprecise Mobile Agents with Firm Deadlines. ETFA 2006 11th IEEE Int. Conf. on Emerging Technologies and Factory Automation. Sep. (2006).

[9] A. Davis, *et al*. Flexible Scheduling for Adaptable Real-Time Systems. Proc. IEEE Real-Time Tech. and App. Symp, May. 1995 (230-239)