A Profiling Based Intelligent Resource Allocation System

J. Monroy, J. A. Becerra, F. Bellas, R. J. Duro, F. López-Peña

{jmonroy, ronin, fran, richard, flop}@udc.es
Grupo Integrado de Ingeniería
Universidade da Coruña

Abstract. The work presented here is mainly concerned with the development of an intelligent resource allocation method specially focused in providing maximum satisfaction to user agents tied to resource strapped applications. One of the applications of this type of strategies is that of remote sensing in terms of energy and sensor usage. Many remote sensors or sensor arrays reside on satellites and their use must be economized, while at the same time the agency managing satellite time would like to satisfy the users as much as possible. Here we have developed a cognitive based strategy that obtains models of users and resource use in real time and uses these models to obtain strategies that are compatible with management policies. The paper concentrates in obtaining the user models.

1 Introduction

The resource allocation problem was initially a matter of concern in a variety of areas in operations research and management science [1]. A wide variety of modern applications concerning electric, electronic, communication, computational or sensor network shared systems are compelled to accomplish two conflicting requirements; on one hand increasing in performance and complexity and, on the other, minimizing resource spending. Thus, all these application require a careful resource allocation and planning methodology. To reach a practical implementation in complex cases, intelligent resource allocation appears as a right tool to optimally allocate and reuse resources (examples may be found in [2], [3], and [4]). An intelligent resource allocation method for the distribution of limited resources based on user profiling and aimed at maximizing user satisfaction is presented in this paper. The method addresses the real time profiling of users integrated in resource assignment processes. It is based on evolution and the capabilities of Artificial Neural Networks (ANNs) to model non linear functions. The main idea proposed here is that the problem of assigning resources to tasks and thus, to users, in environments under constraints where the satisfaction of the users is important, may be solved through the implementation of a cognitive process that allows the decision system to directly model in real time the behavior of the users and their satisfaction and use this information as a fitness measure when deciding a resource assignment strategy.

A cognitive strategy usually involves two different tasks: to use the information from the interaction with the environment in order to obtain appropriate models of the

environment an of the system itself, and to use the model set as an evaluator so that strategies can be tested on it so that the most adequate one may be chosen before applying it. In the particular case of energy strapped systems that are shared by many users, it is necessary to be able to model the behavior of the users and of the system as a response to their requests as well as their satisfaction under different circumstances so that these models can be employed to decide the resource allocation sequence that provides the most satisfaction and at the same time preserves the integrity of the system in terms of energy while respecting management policies.

2 Cognitive approach

One classical way of formalizing the operation of a general cognitive model for a general agent from a utilitarian point of view starts from the premise that to carry out any task, a motivation (defined as the need or desire that makes an agent act) must exist that guides the behavior as a function of its degree of satisfaction. The tools the agent can use to modify the level of satisfaction of its motivation are perceptions through sensors and actions through actuators. In a Cognitive Mechanism, the exploration of actions must be carried out internally and thus a set of mathematical functions, usually denoted as W and S must be somehow obtained. These functions correspond to what are traditionally called:

• *World model (W):* function that relates the external perception before and after applying an action, which in this cases is related to the behavior of the user.

• *Satisfaction model (S):* function that provides de predicted satisfaction from the predicted external perceptions provided by the world model.

There are two processes that must take place in a real non preconditioned operating mechanism: models W and S must be obtained as the agent interacts with the world, and for every interaction of the world, the best possible action must be selected through some sort of optimization using the models available at that time.

When trying to implement this cognitive model computationally, in addition to these basic elements (perceptions, actions, motivations and models) we need to include a new one: the *action-perception pair*. It is just a collection of values from the interaction with the environment after applying an action, that is, data from the real world corresponding to the sensorial data before applying an action, the action applied, the sensorial data after the application of the action and the satisfaction achieved. This action perception pair is a representation of data from the real world and can thus be used to evaluate models in real time.

3 Constructing the MDB

As we have mentioned in the previous section, the actions that must be applied in the environment are selected internally by the agent and this functionality comprises three main elements: a memory that stores the action-perception pairs, a stage to improve the models according to the real information available and finally a stage to select the action to be applied.

Using this scheme we have constructed a Cognitive Mechanism called MDB (Multilevel Darwinist Brain) [5]. The main difference of the MDB with respect to other model based cognitive mechanisms is the way the models are obtained and the actions planned from them. Its functional structure is shown in Fig. 1. The final objective of the mechanism is to provide the action the agent must apply in the environment to fulfill its motivation. Its operation can be summarized by considering that the selected action (represented by the Current Action block) is applied to the Environment through the Actuators obtaining new Sensing values. These acting and sensing values provide a new action-perception pair that is stored in the Short-Term Memory. Then, the Model Learning processes start (for world and satisfaction models) trying to find functions that generalize the real samples (action-perception pairs) stored in the Short-Term Memory. The best models in a given instant of time are taken as Current World Model and Current Satisfaction Model and are used in the process of Action Optimization. The best action obtained from this process is applied again to the Environment through the Actuators obtaining new Sensing values.

These five steps constitute the basic operation cycle of the MDB, and we will call it an iteration of the mechanism. As more iterations take place, the MDB acquires more information from the real environment (new action-perception pairs) so the models obtained become more accurate and, consequently, the action chosen using these models is more appropriate.

There are two main processes that must be solved in MDB: the search for the best world and satisfaction models predicting the contents of the action-perception pair memory and the optimization of the action trying to maximize the satisfaction using the previously obtained models. In the way these processes are carried out lies the main difference of the MDB with respect to other cognitive mechanisms.

In this context, a model is just a non linear function defined in an n-dimensional space that approximates and tries to predict real characteristics. The techniques for obtaining these functions must be found considering that we have samples (action-perception pairs) of the function to model, these samples are known in real time and we want to obtain the most general model possible, not a model for a given set of samples present in a particular instant of time.

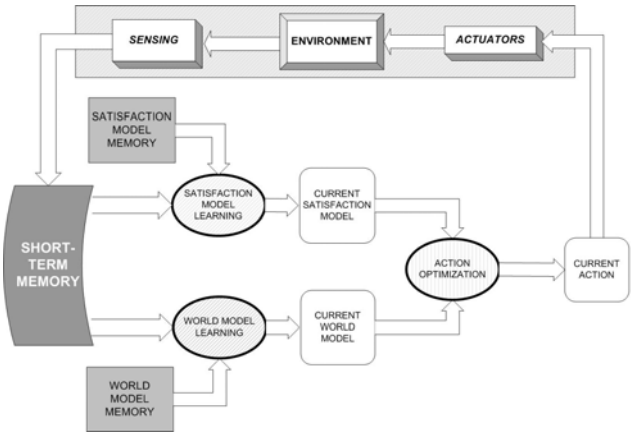

Fig. 1. Block diagram of the MDB

Taking these three points into account, the model search process in the MDB is not an optimization process but a learning process. As commented in [6], learning is different from optimization because we seek the best generalization, which is different from minimizing an error function. Consequently, the search techniques must allow for gradual application as the information is known progressively and in real time. In addition, they must support a learning process through input/output pairs (action/consequence samples) using an error function.

To satisfy these requirements we have selected Artificial Neural Networks as the mathematical representation for the models and Evolutionary Algorithms as the most appropriate search technique. This combination permits an automatic acquisition of knowledge (the models), providing a Darwinist base to the MDB as explained in [5].

After applying an action in the environment and obtaining new sensing values, the search for the models are now evolutionary processes, one for the world models and another for the satisfaction models. The use of evolutionary techniques permits a gradual learning process by controlling the number of generations of evolution for a given content of the action-perception pair memory. This way, if evolutions last just for a few generations (usually from 2 to 4) per iteration, we are achieving a gradual learning of all the individuals. Furthermore, the evolutionary algorithms permit a learning process through input/output pairs using as fitness function an error function between the predicted values provided by the models and the expected values for each action-perception pair

4 Application of the MDB to this particular problem

In this particular problem there are two entities for which we must obtain models, the users of the system and the resources of the system. As a first approximation, in this paper we are going to concentrate on the use of the MDB through the production of user behavior models and consider the rest of the elements as fixed.

A user model is implemented as an Artificial Neural Network based agent that processes information corresponding to the amount of resources the user requests, some data on the day of the week and the time of day the resources are requested. As outputs these models provide an estimation of the final use of resources the user really made. This data can then be used by the strategy optimization stage in order to provide the best sequence of tasks in order to maximize the satisfaction of the users while respecting the energy requirements and constraint of the system.

As an experiment to show the way the system works, we have constructed a process queue managing system that contains one model per user and implements a FIFO like short term memory. The user models are ANNs with four input nodes (requested length of process, walltime, day requested, time of day the request was made), two hidden layers with 3 nodes each and an output layer with two nodes (estimated real length of process and estimated walltime).

The user models are updated every time a user requests a process. Initially there is a set of randomly generated models for each user. When the user requests resources, the input data and the results of the running of the processes are used to run a few evolution generations. This evolution uses the corresponding model set and has the aim of minimizing the error between the values predicted by the model and those that were obtained. These evolution

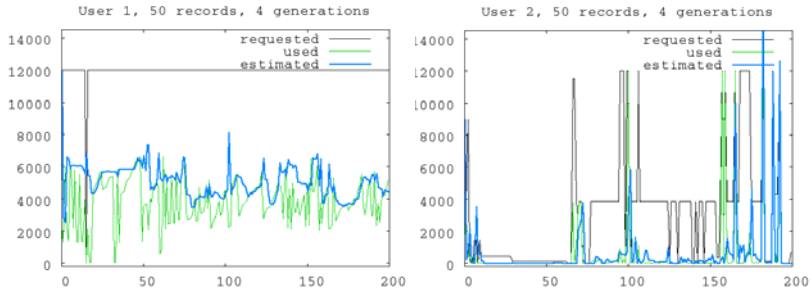

Fig. 2. Some results for two users selected at random. The graphs display the amount of requested resource (in this case computing time), the estimation made by the ANN based user model and the real usage of the resource.

processes consist of a standard genetic algorithm with tournament selection (2% window), one point crossover, random mutation and elitist replacement strategy. The best model for each user in a given instant of time is defined through a fitness criterion based on its estimation of the behavior of the user in the previous instants of time. We want the model to provide a good estimation, but we favor overestimation (we do not want to cut a process short) through a non linear fitness function.

The best model for each user in a given instant of time (the model that best predicted the previous task request parameters) is used as the estimator for the strategy optimization block to obtain the most appropriate strategy. This block takes into account all of the requests that are active and uses the user models corresponding to these requests to predict the real resource requirements for each user. With these predictions and considering a fitness criterion that involves user satisfaction through user satisfaction models or management policy rules, and the resource energy constraints the strategy optimization block provides a queue of processes to be run on the system. Once these processes are run, the results obtained can be used to improve the models.

Initially the system will provide poor results as the models are basically random. As time passes and more information is considered, the models will improve and thus, their predictions will be more adequate, leading to a better planning by the system.

User	10, 4	10, 10	10, 50	20, 4	20, 10	20, 50	50, 4	50, 10	50, 50
1	53.989 0.270 0.141 0.375	56.379 0.282 0.142 0.377	75.095 0.375 0.195 0.442	43.570 0.218 **0.103** 0.321	51.600 0.258 0.130 0.459	75.704 0.379 0.209 0.457	43.432 **0.217** 0.107 0.326	59.553 0.298 0.160 0.400	67.275 0.336 0.193 0.440
2	132.258 0.661 0.554 0.744	120.107 0.600 0.527 0.726	141.994 0.710 0.601 0.775	114.258 **0.571** 0.550 0.742	126.155 0.631 0.551 0.742	143.554 0.718 0.592 0.769	121.040 0.605 0.526 0.725	124.489 0.622 **0.518** 0.720	137.541 0.688 0.587 0.766
3	84.056 0.420 0.344 0.587	87.307 0.437 0.379 0.613	91.172 0.456 0.406 0.637	79.582 0.400 0.346 0.590	77.035 **0.386** 0.346 0.589	87.347 0.437 0.383 0.619	77.330 0.387 **0.308** 0.555	84.216 0.421 0.360 0.600	89.685 0.448 0.399 0.632

Table 1: Some results for three of the users and different STM sizes and number of generations of evolution per interaction with the World (STM size and generations in the heading of each column). The data in each cell is the sum of the error, its average, variance and standard deviation. Shaded cells are best results for each user.

5 Experiments

In what follows we present some results of the performance of user models obtained from real data in a resource allocation problem. The users in this problem request computing time and resources up to a maximum allowed value and the system must obtain the best possible allocation strategy in order to maximize resource use and user satisfaction.

The experiments were carried out modeling a number of different users and using different parameters for the MDB, the main two parameters considered were Short Term Memory Length (10, 20 and 50 registers. In all cases the STM behaved as a FIFO) and number of evolution generations for each real action perception pair obtained (4, 10 and 50).

Figure 2 displays some results for two users selected at random. The graphs display the amount of requested resource (in this case computing time), the estimation made by the ANN based user model and the real usage of the resource. This graphs show how the system is able to obtain quite appropriate models of general resource use by a given user in real time, even in cases where the behavior of the user seems to change quite randomly. In fact, when a pattern of behavior changes brusquely, the system lags a little behind in its predictions, but soon it captures user behavior.

In table 1 we display a set of results for different number of registers in the STM and different number of evolution generations in between interaction with the world. Several aspects can be mentioned. The first one is the improvement achieved with the increase in STM size. Only in one case (second user) the average is worse for a STM size of 50 than for a size of 20, although the variance is still better. However, an increase in short term memory size implies an increase in the computational cost. Consequently equilibrium between these two conflicting aims is desired. In the example presented here, this equilibrium point could be established at 20 as for this value, the results are almost as good as for an STM length of 50 in every case. This fact can be appreciated in Fig. 3 where these two cases are compared for user 3 and the differences are quite slight.

On the other hand, a smaller the number of generations in between interactions with the world leads to better the results. A large number of generations of evolution in between interactions leads to a model that fits the particular contents of the STM in that instant of time but does not generalize well, resulting in larger errors for new estimations. Consequently, it

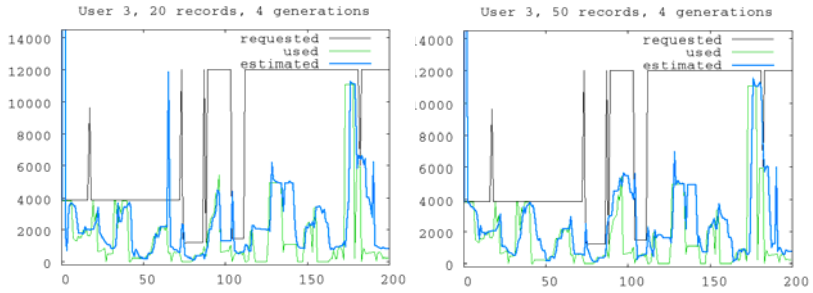

Fig. 3. Some results for obtained for the same user using a STM of 20 records and a STM of 50 records. The graphs display the amount of requested resource (in this case computing time), the estimation made by the ANN based user model and the real usage of the resource.

seems that for computational reasons, the best bet is to choose very few generations of evolution in between interactions with the world, between 2 and 4 and a not very large STM of around 20 registers. Notwithstanding these comments, if one compares requested time and estimated time, it is clear that by using this methodology, the profiles obtained would clearly help to improve the planning of the processes.

Conclusions

An evolutionary agent based strategy through the incorporation of user profiling is proposed for the optimal assignment of energy strapped limited resources that must accommodate multiple users and require their satisfaction. This strategy provides a mechanism for taking into account the real behavior of the users in terms of resource usage as opposed to resource request and their associated satisfaction when planning the use of the systems.

The approach is based on an on line evolutionary mechanism that adapts sets of artificial neural networks that model the users through their interaction with the world. The adaptation of the models is carried out through a few steps of evolution every time the system interacts with the world, thus leading with time to good models of the real behavior and satisfaction of the users which provide a base for an optimization of resource usage in terms of these models and the policies established by the controlling agency. The results show that the models obtained really improve on the requests made by the users in a problem taking into account elements such as the day the request was made, the time it was made and the amount of resource required as well as its walltime.

Acknowledgements

This work was funded by the MCYT of Spain thorugh project VEM2003-20088-C04-01 and Xunta de Galicia through project PGIDIT03TIC16601PR.

References

1. Ibaraki, T. “Resource Allocation Problems”. Algorithmic Approaches. MIT Press, 1988
2. Quijano, N., Gil, A. E. and Passino, K. M. “Experiments for Dynamic Resource Allocation, Scheduling, and Control”. IEEE Control Systems Magazine; 63-79, 2005
3. Al agha, K. “Resource Management in Wireless Networks using Intelligent Agents”. Int. J. Network Mgmt 2000; 10:29–39, 2000
4. Bhaskaran, S. Forster, B. Paramesh, N. Neal, T. “Remote sensing, GIS and experts systems for fire hazard and vulnerability modelling and dynamic resource allocation”. IEEE Geoscience and Remote Sensing Symposium, IGARSS '01. 795-797 vol.2, 2001
5. Bellas, F., Duro, R. J. “Multilevel Darwinist Brain in Robots: Initial Implementation”, ICINCO2004 Proceedings Book (vol. 2), pp 25-32, 2004.
6. Yao, X., Liu, Y., Darwen, P., “How to make best use of evolutionary learning”. Complex Systems: From Local Interactions to Global Phenomena, pp 229-242, 1996.