SELF-ORGANISED DISTRIBUTION OF TASKS INSIDE A NETWORKED ROBOTIC SYSTEM

Sebastian Smolorz and Bernardo Wagner
Institute of Systems Engineering, Real Time Systems Group
Leibniz Universität Hannover, Appelstr. 9A, D-30167 Hannover, Germany

Keywords: Autonomic Computing, Organic Computing, Self-organisation, Self-X, Real-time, Optimisation, Genetic algorithm, Task distribution, Networked robotic system.

Abstract: For several years, mobile multi-robot systems are a main focus in robotic research. The possibilities to carry out demanding algorithms in a networked robotic system grow as computational resources used by those robots become more and more powerful and inexpensive. One major question is how to distribute the necessary algorithms and tasks to the available computers so that all requirements are met and furthermore, an optimal distribution pattern is achieved. This paper presents a new approach for generating an optimal task distribution pattern without human intervention based on the concepts of Autonomic Computing and Computational Intelligence. Although still in its conceptual state, we describe why our approach seems to be promising for this particular problem. Additionally, further challenges and research demand are outlined.

1 INTRODUCTION

Autonomous mobile robots used for research during the past decade usually were systems with only one computer taking care of executing all software modules, computations and algorithms. In the field of robotic research, several algorithms exist that have high computational costs, like the ICP algorithm used for 3D point mapping as stated by Langerwisch and Wagner (2010). Those algorithms and with it the quality of the robots' application would benefit directly from the availability of more computation power, especially when real-time constraints have to be considered. One viable way to boost performance is to connect several computers and parallelize the algorithmic load.

Advanced frameworks for robotic applications are usually organised in modules with well-defined responsibilities and interfaces (Wulf et al., 2003). By using a messaging service to exchange data via a real-time capable network like RTnet (Kiszka et al., 2005), a distribution of interdependent tasks with fulfilled real-time requirements can be achieved.

The proper distribution of tasks to computers is not always apparent, especially when a large number of tasks and computers exist. Certain requirements have to be fulfilled in any case, like deadlines. But possible distribution patterns could differ in quality, like the used network bandwidth or the load of computers. To find a satisfying distribution pattern by hand can be tedious at least; it could become impossible at a high number of tasks and computers.

This paper presents the proposal of a system capable of finding an optimal distribution pattern for a set of real-time tasks and computers automatically. Based on the concepts of Autonomic Computing (Kephart and Chess, 2003) and Organic Computing (Schmeck, 2005), the proposed system distributes the tasks in an optimal manner with regard to the minimal usage of network bandwidth and the algorithmic load balance. The system is adaptive and dynamic, i.e. it does not only emit a valid distribution pattern during the setup phase but handles environmental changes while the networked robotic system performs like loss of processing capacity or added computing components.

Section 2 gives a short overview of related work and existing approaches in this area. In Section 3 the concept of the proposed system is introduced together with the idea of the optimization approach. Section 4 outlines the open questions and challenges that lie on the way to a real robotic system with the presented autonomic properties. In Section 5 the conclusion of our idea of a self-organised networked robotic system is given.

Smolorz S. and Wagner B..
SELF-ORGANISED DISTRIBUTION OF TASKS INSIDE A NETWORKED ROBOTIC SYSTEM.
DOI: 10.5220/0003572402350238
In *Proceedings of the 8th International Conference on Informatics in Control, Automation and Robotics* (ICINCO-2011), pages 235-238
ISBN: 978-989-8425-74-4

2 STATE OF THE ART

The question of implanting a software component with self-management capabilities into a real-time robotic control system was posed by Steiner, Hagner and Goltz (2007), especially for the control of parallel kinematic machines. One goal the authors wanted to achieve with the aid of this so called self-manager was the distribution of control tasks to several connected PCs in a way that higher cycle frequencies of control components became possible. The scheduling analysis tool SymTA/S (Henia et al., 2005) was used to check whether potential distribution patterns meet all real-time requirements before actually applying them.

The drawback of this approach was the necessity of creating SymTA/S models of every system before the execution could start. In contrast, Stein, Hamann and Ernst (2006) present a first step towards a distributed online performance analysis and optimization system. However, SymTA/S models are not generated automatically during the execution of a distributed system but supposed to be delivered together with system or application changes.

The facility to distribute tasks to several computing units by means of a self-manager leads to the question how an amount of tasks is apportioned. Steiner et al. (2008) state that an optimal mapping of control tasks to control PCs minimizes the network communication and balances the algorithmic load. As there are M^N possibilities to distribute N tasks to M computing units a two-stage heuristic is proposed to tackle this NP-hard problem. The first stage arranges for the balancing of the algorithmic load while the second step tries to minimize the bus communication. This last step is prone to destroy some of the balance reached in the first stage so it could be seen as problematic in view of the whole optimization problem.

Jakob et al. (2009) broached the issue of job scheduling and resource allocation in the context of grid computing which can be seen as related problem. Users send jobs to a grid cluster and expect them to be executed as quickly as possible but also preferably cheap. The provider of the cluster, on the other hand, is keen to operate the resources at full capacity and for this reason generate more income. The authors introduce an Evolutionary Algorithm (EA) to optimise schedules which were planned by a previous heuristic. Experiments show measurable improvements over the heuristic plans, even correcting illegal schedules because of budget violation.

3 CONCEPT

Steiner et al. (2008) identify two criteria which act as optimization targets for the distribution problem:

1. Minimizing network communication;
2. Balancing the algorithmic load.

These criteria are adopted in our concept since they lead to reasonable task distribution patterns. Minimizing communication over a slow bus improves the response time of interconnected modules as their data can be copied inside the fast system memory. The more bandwidth remains free the easier it gets to add more data transmissions between the computers if it becomes necessary at a later time during system operation. A small bandwidth utilization lowers the risk of a malfunctioning system in case of fragile network quality.

Balancing the algorithmic load is desirable if an algorithm can be scheduled in parallel. A good example is the Monte Carlo Localization used by mobile robots to determine their global position inside a given map (Wulf et al., 2005).

Choosing an appropriate optimization algorithm is crucial for obtaining the optimal distribution pattern for a given task-resource conglomerate in a specific situation. In opposition to Steiner et al. (2008) we propose a genetic algorithm (GA) to tackle the distribution problem. GAs, or EAs generally (Engelbrecht, 2007), have strengths when it comes to multi-objective optimization problems. Moreover, GAs are not limited to a certain problem class, thus being applicable to our distribution problem in principle. However, for a successful optimization process the genetic problem representation plays an essential role which will be handled in the following subsection in more detail.

3.1 Optimization Algorithm

The chromosome of the problem representation must be suitable to conduct the usual genetic operations (selection, crossover, mutation) and permit an easy interpretation of the genotype, in our case the mapping of tasks (T_n) to computing resources (CR_m). We propose a chromosome containing as many genes as tasks are intended to be distributed. The value of one gene, the *allele*, determines the computing resources the task is executed on.

Figure 1 shows an exemplary initial population of three individuals. Six tasks (T_1-T_6) are to be distributed to three computing resources (CR_1-CR_6). For example, the second individual of the initial population assigns T_3 and T_6 to CR_1, T_1 and T_4 to

CR_2 and T_2 and T_5 to CR_3. Figure 1 also shows a standard GA process with individual 1 and 3 being selected and crossed after gene 4 of individual 1. A mutation is missing in this example but not excluded in general. More about the fitness evaluation follows in Subsection 3.4.

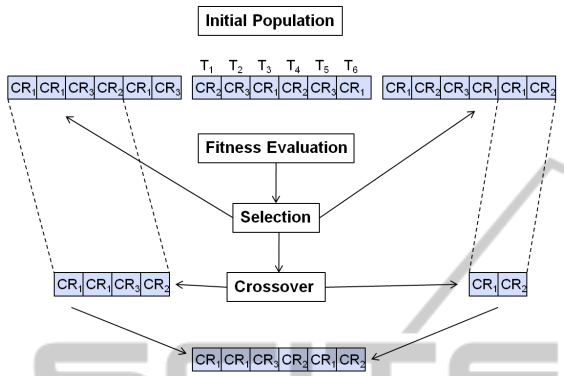

Figure 1: Genetic Algorithm.

3.2 Initial Task Distribution

Before putting a networked robotic system into operation the decision about a correct and optimal distribution pattern has to be made, thus it is necessary to let the evolution process run in advance. This setup mode is noncritical and lets the robotic system perform different task distribution patterns to test the validity and quality of the pattern with respect to bus communication and algorithmic load distribution. Figure 2 depicts the setup mode and the subsequent operating mode:

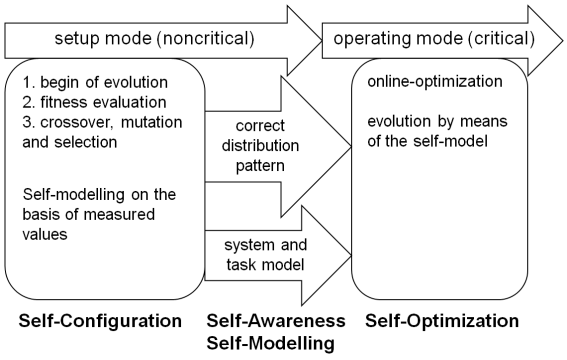

Figure 2: Setup and operating mode.

The gathering of relevant system data measured during the live runs of different distribution patterns can be used to construct a model of the whole networked system. This self-model can be used later to perform further evolution steps without interfering with the operating system.

Once the setup mode is completed, i.e. a correct distribution pattern is found and the evolution process has reached an individual good enough for putting the robotic system into operation self-configuration (in terms of Autonomic Computing) was successful and the system enters the operation mode. This mode is critical since now it is not feasible to examine further distribution patterns. Nevertheless, it is desirable to continue evolution because there might exist a better distribution pattern. This self-optimization can be carried out with the help of the self-model in parallel to the normal tasks of the system.

Once a better individual than that forming the current distribution pattern is found a reconfiguration of the task arrangement can be conducted when the system enters a safe system state. The construction and utilisation of a self-model can be seen as a form of self-awareness.

3.3 Error Status

Due to external conditions computing resources could become unavailable, e.g. in case of malfunctioning hardware, during the runtime of the system. In such a situation the tasks being executed on the now missing resource have to be redistributed to the remaining resources. The system has to enter an error status, accompanied by an emergency halt of all critical components. Since an unknown distribution pattern with less alleles has to be found a new evolution run with a transition to the setup mode is necessary. Figure 3 illustrates this self-healing procedure.

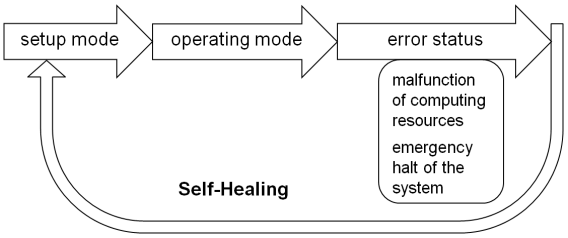

Figure 3: Error Status (Self-Healing).

3.4 Fitness Evaluation

During the evolution process individuals with illegal distribution patterns could arise, e.g. with deadline violations. They could be detected during the setup phase and sorted out. The fitness of correct distribution patterns has to be evaluated according to the objectives specified above. Since minimizing the bus communication is contrary to balancing the algorithmic load it has to be analysed how those

conflictive objectives can be both optimized. We have to check which multi-objective optimization approach is applicable for our problem. One general approach was described by Beume et al. (2008).

4 FUTURE WORK

The described concept still has to be proven and tested. Preparations for experiments are under way, an adequate software framework for mobile real-time robotic systems with a plethora of modules and sophisticated algorithms is available (Web, 2011). A good candidate for gathering first experimental results is the Monte Carlo Localization (MCL) as described in Section 3. In addition to the need for parallel computation resources during global localization, the position tracking of the MCL needs high iterative computing power. Therefore we have to consider that situations exist where balancing the algorithmic load is not an appropriate evolution objective but rather a minor loaded computing resource to carry out a high number of algorithmic iterations.

The autonomic self-manager which is responsible for finding distribution patterns by means of the presented Genetic Algorithm as well as constructing the self-model and online-optimization has to be implemented and integrated into the robotic framework RACK (Web, 2011). An open issue is the way the online-optimization will be realized. We have to evaluate whether the approach of Stein et al. (2006) (see also Section 2) is viable for us or whether the scheduling analysis algorithms SymTA/S is based on (Henia et al., 2005) will be directly integrated into our self-manager.

5 CONCLUSIONS

In this paper, we presented the idea of a system capable of autonomously distributing tasks to available computing resources. The system will be able to generate correct and optimal distribution patterns in order to set up the system. During this procedure a self-model of the system is obtained and used while the system operates to further optimize the distribution pattern currently in use. We described how the system performs self-healing in case of failing components. The need for more research in the field of the proposed optimization approach as well as future work for getting experimental results were outlined as well. The system is well suited for mobile autonomous networked robots which are employed for various duties in complex environments where localization algorithms are used as well as alternate requirements cause differing calculation times of the involved algorithms.

REFERENCES

Beume, N., Naujoks, B. and Rudolph, G. (2008). SMS-EMOA – Effektive evolutionäre Mehrzieloptimierung. *at – Automatisierungstechnik*, 7/2008, pp. 357-364.

Engelbrecht, A. P. (2007). *Computational Intelligence: An Introduction* (2nd ed.). Wiley.

Henia, R., Hamann, A., Jersak, M., Racu, R., Richter, K., Ernst, R. (2005). System level performance analysis – the SymTA/S approach. In *IEEE Proc. Computers and Digital Techniques*, Vol. 152, 2, pp. 148-166.

Jakob, W., Hahnenkamp, B., Quinte, A., Stucky, K.-U. and Süß, W. (2009). Schnelles Scheduling mit Hilfe eines hybriden Evolutionären Algorithmus. *at – Automatisierungstechnik*, 3/2009, pp. 106-114.

Kephart, J. O. and Chess, D. M. (2003). The Vision of Autonomic Computing. *IEEE Computer*, 36(1) :41-50.

Kiszka, J., Wagner, B., Zhang, Y. and Broenink, J. (2005). RTnet – A flexible hard real-time networking framework. *10th IEEE International Conference on Emerging Technologies and Factory Automation*, Catania.

Langerwisch, M., Wagner, B. (2010). Registration of indoor 3D range images using virtual 2D scans. *7th International Conference on Informatics in Control, Automation and Robotics*, Funchal, Madeira, Portugal.

Schmeck, H. (2005). Organic Computing – A new vision for distributed embedded systems. *Proc. of the 8th IEEE International Symposium on Object-Oriented Real-Time Distributed Computing*, pages 201-203.

Stein, S., Hamann, A. and Ernst, R. (2006). Real-time property verification in Organic Computing Systems. In *Proceedings of the 2nd International Symposium on Leveraging Applications of Formal Methods, Verification and Validation*, Paphos, Cyprus.

Steiner, J., Hagner, M., Goltz, U. (2007). Runtime analysis and adaption of a hard real-time robotic control system. *Journal of Computers*, Vol. 2 (10), pp. 18-27.

Steiner, J., Amado, A., Goltz, U., Hagner, M. and Huhn, M. (2008). Engineering self-management into a robot control system. *Proc. of 3rd International Colloquium of the Collaborative Research Center 562*.

Web (2011). The Robotics Application Construction Kit – RACK. *http://developer.berlios.de/projects/rack/*.

Wulf, O., Kiszka, J., Wagner, B. (2003). A compact software framework for distributed real-time computing. In *5th Real-Time Linux Workshop*, Valencia, Spain.

Wulf, O., Khalaf-Allah, M. and Wagner, B. (2005). Using 3D data for Monte Carlo Localization in complex indoor environments. In *European Conference on Mobile Robots (ECMR)*, Ancona, Italy.