

# **A Study on Self-Organization in a Production Environment**

*Elpida S. Tzafestas*

## **1 Introduction**

Modern cellular manufacturing systems involve distributed environments of robots, machines, materials etc. The principal feature of such a system is that incomplete or imperfect information is given about the input flows and the interactions between parts of the system, so that it has to be adaptive to variations in input and environmental conditions. An additional wish for a cellular manufacturing organization is to be as easily reconfigurable as possible, i.e. as easy to adjust to different environments of the same type as possible [1,2,3,4,5]. For all those purposes, the system has to self-organize in order to respond to the inputs while meeting the operational constraints involved. Self-organization in a distributed environment is achieved through the use of fairly simple local, adaptive behavioral rules that have to be chosen so as to optimize if possible the overall system's operability, as defined by a set of objectives and/or constraints [6,7,8].

Our long-term goal is to study and understand the nature and specificities of self-organization by examining cellular manufacturing case studies, which, unlike other self-organization environments, define clear goals and operational criteria. In what follows, we will adopt the medium term goal of exploring the notion of behavioral persistence and task specialization in such a case study. The manufacturing environment and the basic behavioral algorithm will be presented

in section 2, while the comparative simulation results for persistence and specialization in various environmental setups will be given in section 3. Finally, section 4 will conclude the paper and describe briefly future directions of research.

## 2 A Case Study

We assume [9,10] an industrial plant that consists of four service units (for instance, assembly cells) that issue demands for tool use. A tool-manager robot goes from unit to unit responding to those demands (Fig. 1). Each service unit increments demands periodically up to an upper limit steady state (which corresponds to a unit demanding maximal tool use). Beyond this point, the unit's demand does not increase anymore (the production of this unit is temporarily stopped). Obviously, this arrival process alone tends to saturate the system (all units tend to maximum demand). Our goal at this stage is to find suitable reactive robot behavioral specifications that will delay this saturation as long as possible, so as to maximize service units' production (assuming that the robot can't be quick enough to service all units at no time). Demands represent service times. The robot stands initially in its rest place in the center; it repeatedly goes to a unit, services it and returns to the center. Each move from the center to a unit or vice versa needs  $T_M$  time units, while each service lasts at most  $T_S$ . Unless otherwise stated,  $T_M=10$  and  $T_S=3$  and all the results reported are averages over 10 runs, each lasting until demand exhaustion after an initial phase of 1000 time cycles, during which demands are issued.

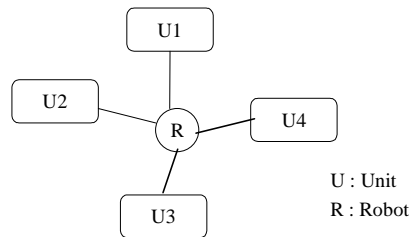


Figure 1. The plant organization

Based on previous experience with reactive robots, we have created one service task for each unit. The tool-manager robot's actuators systems repeatedly evaluate those tasks' motivations and "choose" for execution the one that expresses maximal preference. For this to work, all maximal demands have been normalized to a value  $d_{max}$ , which for the present experiments has been set to 40. Each task's motivation is a function of the corresponding unit's demand, such that when demand = 0, motivation = 0. This way, the robot's "nirvana" is when all its motivations are 0; every time its motivations are non-zero the robot tries to

act in a way that will bring them back to 0. The presence of the process, that tends to saturate those demands, perturbs the motivational state of the robot and causes it to act/react. The robot control programs are written using a cellular control architecture [10,11], while the actual high-level pseudo-code run by each service task is:

- If the robot is at the corresponding service unit, service it for a computed amount of time.
- If the robot is at center, go to service unit.
- If the robot is elsewhere, go to center.

The tasks' motivations are being evaluated after every such step (move or service). It should be stressed that the navigation and perception systems of the robot have been written once and for all in the beginning and never changed since. Sole intervention in the motivational level of each task has allowed drastic increases in the performance to be observed, as will be shown below.

Note that this is *not* a usual resource allocation problem (see, for example, several papers in [12]): we assume that the individual service units possess a local inventory of tools that they use at will and there is an additional set of tools that need to be shared across units and whose use is sporadic. The tool-manager robot regulates the units' demands for those more "rare" tools, which it is more cost-effective to share rather than possess local instances of them. For this to work, we also assume that individual service units make optimal local use of those tools, for instance by regrouping in time operations that depend on these tools. Such a degree of local intelligence in the service units enables us to afford a tool-manager robot that will decide alone when to retreat from a given unit, in which case it will withdraw all tools previously allocated to it, to make them available to other units. Therefore, the effective operation of the units-manager system depends on the existence of such a cooperative coupling. We can certainly proceed to more complex allocation cases, however, by only defining different motivations for different instances of tools that will be therefore treated and allocated independently.

The best motivational rule found so far, i.e. the motivational rule that optimizes the system's performance, is multiplicative and is applied only to the selected task [9].

$$\text{motivation} = \text{demand} * \text{persistence\_factor}$$

Obviously, the persistence factor should be greater than 1. In the worst case (when all service units are saturated with their maximum demand value  $d_{\max}$ ), it makes sense to have the robot satiate units in turn. In that case, a selected task should persist for as long as its demand is non-zero, hence  $\text{persistence\_factor} > d_{\max}$ . To write the corresponding cellular control program, however, we needed a continuous function for the persistence factor, so we adopted the exponential time-convergent

$$\text{persistence\_factor} = 1 + d_{\max} (1 - e^{-t/T_p})$$

which rises from 1 to  $d_{\max}+1$  ( $t$  is the time that has elapsed since selection of the task). Since this rise should happen almost immediately after a task is selected, the  $T_p$  constant should be such that  $1+d_{\max}(1-e^{-1/T_p})>d_{\max}$ , which gives  $T_p < 1/\ln(d_{\max})$ . For the present experiments we used a safe value of  $T_p$  approximately equal to  $1/2\ln(d_{\max})$ ,  $T_p = 0.13$  (since  $d_{\max} = 40$ ).

### 3 Simulation Results

The process has been simulated and studied for multiple robots, in search of self-catalytic models of specialization, that would make each of the robots specialize explicitly (permanently) in a fraction of the service units. Self-catalysis means that the motivation of a task depends on a term that is reinforced on execution of the task. Firstly, a number of experiments have been conducted with additive, multiplicative, multiplicative with an upper bound and self-regulated factors. The second case was far superior to the first, but motivational factors could rise to a very high numeric value; definition of an upper bound solved this problem but resulted in low performance and deadlocks. Self-regulation instead of self-catalysis of the specialization factor has yielded the best results for multiple robots, but still only marginally better than the equivalent case of multiple robots with only persistence. This hardly detectable improvement of the performance with specialization compared with sole persistence can be attributed to the asynchronicity of the system's operation which induces, for example, slightly better regulation of a single-robot 2-unit plant than a 4-unit one that is managed by two robots. The specialization model by self-regulation of the corresponding bias factor is given below :

$$\text{Motivation} = \text{demand} * \text{persistence\_factor} * \text{specialization\_bias}$$

with

$$\begin{aligned} \text{Bias}(t) &= \text{Bias}(t-1) + r * (\text{Bias}_{\max} - \text{Bias}(t-1)), \\ \text{Bias}_{\max} &= 1 \text{ and all biases are normalized according to } \sum \text{bias}(t) \end{aligned}$$

We then proceeded to examine the case of interdependent service units where the output of one unit is fed as input demand to one or more other units, so that a flow network is built (such as in Fig. 2). This study was prompted by the observation [9] that specialization as studied above is probably not beneficial because service units are independent and no genuine "load balancing" is necessary in the system. The network structures studied were multi-level structures with full or 1-1 connectivity between levels. To our surprise, and while the self-regulated specialization mechanism has been found to work correctly (for example, in the case of 3 robots and 9 units, each robot quickly

specializes to three of the units), there has been no quantitative improvement to the performance of the overall system.

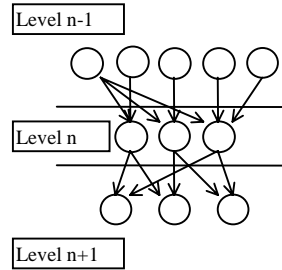


Figure 2. A network of connected units

Next, we studied the case of the initial 4-unit “flat” setup with variable  $T_M$  across units and 2 robots. An indicative setup and the corresponding results are given in Tables 1 and 2.

	$U_1$	$U_2$	$U_3$	$U_4$
$T_M$	6	8	10	12

Table 1. A variable  $T_M$  setup for the 4-unit flat plant

Model	Completion time
<i>Persistence only</i>	1052.25
<i>Specialization</i>	1052.3

Table 2. Results in the case of the plant of table 1

As is obvious from the tables, no significant difference exists between the results without and with specialization. This qualitative property has been verified for various  $T_M$  setups and is maintained even is the  $T_M$  factor is introduced in the initial persistence law :

$$\text{persistence\_factor} = (1 + d_{\max}(1 - e^{-t/T_p})) * \text{lcm}(T_M, \text{ for each unit}) / T_M$$

Next, we studied the standard plant setup ( $T_M=10$ , for each unit) with an additional switching cost matrix, in an effort to study the need for specialization whenever switching costs are present and the secondary goal of minimizing the total cost appears. An indicative cost matrix and the corresponding results are given in Tables 3 and 4.

	$U_1$	$U_2$	$U_3$	$U_4$
$U_1$	0	1	2	3
$U_2$	1	0	3	2
$U_3$	2	3	0	1
$U_4$	3	2	1	0

Table 3. A symmetric cost matrix for the 4-unit flat plant

<b>Model</b>	<b>Completion time (cost)</b>
<i>Persistence only</i>	1026.0 (136.30)
<i>Specialization</i>	1026.0 (152.15)

Table 4. Results in the case of the plant of table 3

Given these results, we tried to improve the overall operational costs through the same specialization model, by moderating a unit's motivation by the corresponding switching cost. The modified motivation rule is given below, and the corresponding results may be found in Table 5. It is obvious from the table that while the completion time is similar without and with specialization, the overall cost drops dramatically with cost moderation, as was expected.

$$\text{Motivation} = \text{demand} * \text{persistence\_factor} * \frac{1}{\text{cost}} [* \text{specialization\_bias}]$$

<b>Model</b>	<b>Completion time (cost)</b>
<i>Persistence only</i>	1026.6 (78.25)
<i>Specialization</i>	1027.13 (74.475)

Table 5. Results in the case of the plant of table 3 with cost moderation

Finally, an asymmetric cost matrix and the corresponding results without or with cost moderation in both cases of presence or absence of specialization are given in Tables 6 and 7. Again, there is no significant difference between sole persistence and specialization, while cost moderation achieves lower total costs, even if the improvement is not so pronounced as in the case of symmetric cost tables.

	<b>U<sub>1</sub></b>	<b>U<sub>2</sub></b>	<b>U<sub>3</sub></b>	<b>U<sub>4</sub></b>
<b>U<sub>1</sub></b>	0	3	1	2
<b>U<sub>2</sub></b>	2	0	1	3
<b>U<sub>3</sub></b>	2	3	0	1
<b>U<sub>4</sub></b>	1	2	3	0

Table 6. An asymmetric cost matrix for the 4-unit flat plant

<b>Model</b>	<b>Default</b>	<b>Cost moderation</b>
<i>Persistence only</i>	1024.32 (130.925)	1027.25 (101.75)
<i>Specialization</i>	1025.62 (125.225)	1026.2 (106.225)

Table 7. Results in the case of the plant of table 6, without and with cost moderation

The qualitative result from these tables is that in no case is the specialization model superior to the pure persistence one :

- If the units are connected in some kind of network, specialization does not seem necessary, because there is some lower bound in the completion time and no other operational cost exists.
- If the factor  $T_M$  varies across units, specialization does not seem necessary, because there is no qualitative difference with the default case (uniform  $T_M$ ), so that only the absolute value of the completion time's lower bound changes.
- Finally, if there are variable switching costs between units, specialization is indifferent, while moderation of persistence by cost achieves lower overall costs.

In all these cases, it appears that there is no genuine specialization problem, i.e., there is no need for robots to permanently commit themselves to a subset of all service units. Temporary commitment is achieved automatically by the persistence model, which should include cost moderation whenever there are additional switching costs.

## 4 Conclusion

The above conclusions for our case study are summarized in table 8.

<i>Concept</i>	<i>Description</i>
<b>Operationality</b>	Need for a motivational model with persistence
<b>Persistence</b>	<ul style="list-style-type: none"> <li>• Is based on a worst-case analysis</li> <li>• Incorporates the operational parameters of the agent</li> </ul>
<b>Specialization</b>	Useless, in all of the cases : <ul style="list-style-type: none"> <li>• Units in a network</li> <li>• Variable <math>T_M</math></li> <li>• Variable switching costs (Symmetric or asymmetric cost matrix)</li> </ul> ⇒ There is no specialization problem, or the specialization problem is solved by the persistence model

Table 8. Conclusions

As further work, we plan to study comparatively the symmetric and asymmetric cost matrices, to understand deeper the significant quantitative difference in performance improvement, as shown in tables 4, 5 and 7. We also plan to study adaptive costs, in the sense that the more a task remains active, the less it costs, which captures one of the major features of specialization. Our

longer term objective is to study the specialization problem in completely different manufacturing setups and to transpose it to domains other than the industry.

## References

1. Jones PF 1992 *CAD/CAM: Features, applications and management*, Macmillan Press Ltd., London
2. Ranky PG 1986 *Computer integrated manufacturing - An introduction with case studies*, Prentice-Hall, Englewood Cliffs, NJ
3. Wright PK, Bourne DA (eds) 1988 *Manufacturing intelligence*, Addison Wesley, Reading, MA
4. Björke O 1979 Computer-aided part manufacturing, *Computers in Industry*, (1979):3-9
5. Kusiak A 1990 Manufacturing systems : A knowledge- and optimization-based approach, *Journal of Intelligent and Robotic Systems*, 3:27-50
6. Baldwin KE 1989 Autonomous manufacturing systems, *Proceedings of the 1989 IEEE International Symposium on Intelligent Control*, Albany, NY, September, pp. 214-220
7. Vaario J, Ueda K 1996 Self-Organization in Manufacturing Systems, *Proceedings 1996 Japan-USA Symposium on Flexible Automation*, July, Boston, MA, pp. 1481-84
8. Tzafestas ES 1998 Autonomous agents in cellular manufacturing, *Advances in Intelligent Autonomous Systems*, SG Tzafestas (ed), Kluwer Academic Publishers, pp. 427-438
9. Tzafestas ES 1998 Reactive robots in the service of production management, *Journal of Intelligent and Robotic Systems*, 21(2):179-191
10. Tzafestas ES 1995 *Vers une systématique des agents autonomes : Des cellules, des motivations et des perturbations*, Ph.D. Thesis, Université Pierre et Marie Curie, Paris, December
11. Tzafestas ES 1994 A cellular control architecture for autonomous robots, *Proceedings of the 1994 International Workshop on Intelligent Robotic Systems*, Grenoble, July, pp. 70-79
12. Desrochers AA, Silva M (eds) 1994 *IEEE Transactions on Robotics and Automation, Special Issue on Computer Integrated Manufacturing*, 10(2), April