Micro- and Macro-Level Interactions in Multi-Agent Manufacturing Systems

Olaf Bochmann * Hendrik Van Brussel Paul Valckenaers

{Olaf.Bochmann, Hendrik.VanBrussel, Paul.Valckenaers}@mech.kuleuven.ac.be

Katholieke Universiteit Leuven
Department of Mechanical Engineering, Division PMA
Celestijnenlaan 300B, B-3000 Leuven, Belgium

Abstract

In addition to optimizing the individual goals of the agents, the macro-level implementation has to optimize a global objective corresponding to a performance criterion of the manufacturing system. These two objectives generate tradeoffs and possible conflicts. To regulate these a job of central command will eliminate such problems, steering the behavior of the units towards the global goal. To this end, central command will be viewed as a "market designer" that can set the reward mechanism in the system so as to ensure that by optimizing their own objectives the agents will also optimize the global objectives.

Keywords:

Multi-agent systems, process interaction

1. INTRODUCTION

Decentralized control architectures or multi-agent systems has been suggested for manufacturing control [17] as an alternative to hierarchical systems. These systems are composed of autonomous components or agents (product, resources, orders) which interact with other components and perform their functions without a central control. In general, agents are described as natural or artificial systems that act within and respond to a permanently changing environment and interact with other agents in their desire to pursue certain goals. The concept of an agent includes a "social notion" and the power of this concept is derived from agents interacting with other agents in their problem solving process. A software representation of an agent usually holds a model of its environment, which reflects the current state of the world or the "problem" in which the agent is involved, and some form of capability specifications that are used by the agent in its acting. Implementations of such agents can be simple reactive systems or complex software entities with reasoning and planning capabilities [12, 7], a knowledge base, an inference mechanism and an explicit model of the problem to be solved– including a model of other agents [10].

*O. Bochmann is a PhD student at the Katholieke Universiteit Leuven, Department of Mechanical Engineering, Division PMA.

The decision-making of agents will be based on certain facts the agent derives from its environment (represented as its "beliefs" about the world) and a current internal state it holds about its goals. As agents are "social" entities, they coordinate their activities with other agents. With that, whole societies are employed to work on specific problems. In case of the Stigmergic approach [9], this coordination is completely driven by an information-rich environment containing "signs". This information is sensed by simple reactive agents and influence their behavior. Agents within the stigmergic approach have a sense of "location" and "proximity" regarding their environment, which enables them to sense "local" information, but they do not have any awareness of partner agents (stigmergy is actually based on observations of natural agent systems like insect societies where similar coordination concepts exist). In this case, the agent follows a static individual goal, e.g. finish before due date or maximize utilization rate. Information may also exist implicitly e.g. as an agreement or contract between two or more agents. Then, the agents goal is to fulfil the contract(s) in addition to the static goal and thus it is a dynamic goal. In both cases, the individual goals of the agents should emerge towards a global goal e.g. optimizing the performance criterion of the manufacturing system.

Agents interact with one another in two distinct ways. Micro-interactions take place among a small number of agents, and generally involve explicit modelling of other agents' decision theoretic behavior, and the use of multi-agent machine learning techniques. Macro-interactions take place among large number of agents, and are based on the economic notion of a market mechanism or exchange. Micro interactions employ single-agent decision theoretic techniques such as BN and partially observable Markov decision processes (POMDPs), but extend them to the multi-agent setting. Market mechanisms [2] and stigmergy [16] are carefully crafted to ensure that the individual decision making by the units are synergistic rather than mutually destructive.

The remainder of this paper is organized as follows. The problem definition and representation is given in section 2. Section 3 and 4 present the two different interaction schemes in detail. The combination of both schemas is discussed in section 5. Finally, section 6 presents a summary of the paper.

2. PROBLEM DEFINITION AND REPRESENTATION

In this research the problem of job-shop scheduling (JSP) is considered. The JSP problem [8] is defined by a set M of machines, $M = \{M_1, \ldots, M_m\}$, and a set J of jobs, $J = \{J_1, \ldots, J_n\}$. The ith job, $i = 1, \ldots, n$, is an ordered sequence of operations from the set O of operations, $O =$

$\{u_{ij}\}$, for $j = 1, \ldots, m$ as index in the sequence. Each operation $u_{ij} \in O$ is assigned to a machine $m_{ij} \in M$ and has to be processed on this machine for p_{ij} consecutive time instances.

The problem is to assign a starting time $S_{ij} (\forall u_{ij} \in O)$ to all operations such that the maximum of the completion time $C_{ij} = S_{ij} + p_{ij}$ of all operations $u_{ij} \in O$ is minimized

$$L = \max_{u_{ij} \in O} (S_{ij} + p_{ij}). \tag{2.1}$$

This is the global performance criterion, and it is called *makespan*. There are two types of constraints. The *precedence relationship* $S_{ij} \geq S_{ik} + p_{ik}$ when $u_{ik} \rightarrow u_{ij}$, expresses the operation precedence on the job chains. The *machine constraint* $(S_{ij} \geq S_{kj} + p_{kj}) \vee (S_{kj} \geq S_{ij} + p_{ij})$ is saying that no more than a single job can be processed at the same time on the same machine. $N = |O|$ is the total number of operations.

An exact solution to this problem is NP-hard [6]. The solution to the problem can be represented by a permutation

$$\pi = (\pi(1) + \pi(2), \ldots, \pi(N)) \tag{2.2}$$

of N entities which contribute to defining the problem.

An instance of the problem is associated with a disjunctive graph model [13], the JSP graph. A JSP with a machine set M, a job set J and an operations set O containing N operations can be represented by a graph $D = (V, A, E)$. The graph is composed of $V = O \cup \{u_0\} \cup \{u_{m+1}\}$ nodes, where u_0 is an extra vertex added in order to represent the start and u_{m+1} for the end of the job shop. All other nodes represent an operation. $A = \{(u_{ij}, u_{ij+1})$ if $u_{ij} \longrightarrow u_{ij+1}$ for $J_i \forall i\} \cup \{u_0, u_{1j}\} \cup \{u_{im}, u_{m+1}\}$ is the set of conjunctive directed arcs, and $E = \{(u_{ij}, u_{hj}) \forall i\}$ disjunctive edges. Each arc connects two nodes, if they follow each other. Nodes connected by arcs belong to the same job. Nodes connected by edges belong to the same machine and define the machine constraint.

Figure 1 presents an example graph with three jobs and four operations. Machine M_1 performs operations u_{11}, u_{22} and u_{31}, machine M_2 operations u_{13}, u_{21} and u_{32}, machine M_3 operations u_{12}, u_{23} and u_{33} and machine M_4 operations u_{14}, u_{24} and u_{34}. The first operation of job 1 and 3 as well as the second operation of job 2 have to be performed on machine M_1, the third operation of job 1, the first of job 2 and the second of job 3 on machine M_2, the second operation of job 1 and the third operation of job 2 and 3 on machine M_3 and, finally, the last operation of all jobs on machine M_4.

Material handling resources are assumed to be always available and transportation times are included in processing times.

3. MICRO-LEVEL INTERACTION

On the micro level three types of agents are considered: product agents, resource agents and order agents [17]. From the machine learning perspective, this type of agent is heterogeneous and may be communicative or non-communicative [14]. Heterogeneous agents have different goals, domain models or actions. An order agent might

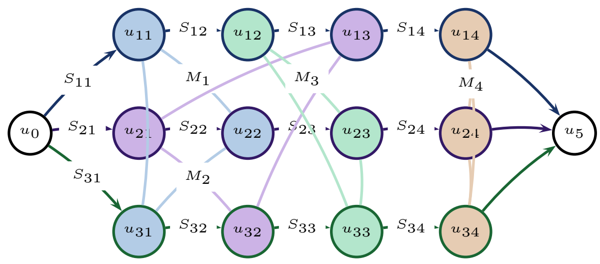

Figure 1: JSP-graph.

have the goal to finish the order before the due date, it therefore models the sequence of operations and acts while making routing decisions or spreading intentions. On the other hand, a resource agent might have the goal to maximize their own utilization rate (minimize setup time), it therefore models the resources schedule and acts while advertising capabilities. Even if they have different goals, they may be friendly to each other's goals (benevolent) e.g. forming an alliance similar to that between an order and a resource agent to allocate a resource. Competitive agents may actively try to inhibit each other. Order agents may compete for the same resource at the same time and try to distract each other.

Non-communicative agents have no means of communicating directly with each other. However, their sensory input and the effects of the actual actions they take differs because they are situated differently in the world. In this scenario, agents can perform simple heuristics. An order agent may sense the processing time for its next operation on different resources and decide to move to the one with the shortest processing time. A resource agent may select from a queue the order with the shortest setup time. An agent might also decide merely to change its location to gain more information. In a scenario of communicating agents, coordination is much more efficient because agents can use direct communication with other agents in addition to the interaction with the environment (sensing, acting). From a practical point of view, this communication might be broadcast, posted on a blackboard for all to interpret, or targeted point-to-point from one agent to another agent. Direct communication is different from an interaction with the environment because here the agent is a special part of the environment, which might be modelled explicitly. Using direct communication, an order agent may communicate to an other order agent that it is approaching a particular resource. The second agent can model this behavior and might decide to head for another resource instead.

A crucial aspect in micro-interactions is the effective representation of utility, beliefs, and the policies of agents [3]. Utility in this context is a complex function that must include capacity of resources, processing time, and amount of time required to meet orders. Non-communicating agents raise several interesting issues in multi-agent systems. Agents might model the internal states, the goals, actions and abilities of other agents. This can be done by observing their actions. Recursive modelling method (RMM) can be used for modelling states [5] and learn models [15] of other agents. Even without direct communication, agents can affect each other by two types of stigmergy, namely active stigmergy and passive stigmergy. Active Stigmergy occurs when agents actively change the environment to affect the sensory input of other agents (spreading pheromone to attract operations to a resource). Passive stigmergy involves changing the environment so that the effects of other agents' actions change. This could be done by changing the evaporation rate of pheromone to alter the effect of this action over time.

Communication among agents raises further interesting issues. For cooperative distributed sensing, techniques such as active sensing [1] can be used. This is the ability to shift attention towards an area of higher uncertainty or interest. An important question is what agents should communicate. In the earlier case they would communicate their sensed states of the world. However, it is also possible to share information regarding the agent's goals. When agents cooperate on a given task for a given time (e.g. resource allocation), they make commitments to each other. A Commitment involves agreeing to a common goal, regardless of whether it serves one's own interest or not. Last commitment is a strategy to keep options open as long as possible to give the agent an opportunity to react to new situations. Commitment in general is needed to make systems run smoothly by providing a way for agents to define for how long and to what extent they are prepared to cooperate [11].

In a non-communicative scenario, agents maintain a synchronous interaction with the environment. In the scope of Bayesian probability theory, agents and Environment perform an input-output information processing activity, where the inputs are states, data, relationships and constraints, that provide information about the system. The outputs are analysis, predictions, and decisions about the system.

The problem can be separated into the following subproblems: modelling, information processing, and decision making.

3.1 Modelling

Bayesian Networks (BNs) are used so that one agent's model includes a representation of its beliefs about factors that influence other agents' decision making processes. Bayesian probability represents and quantifies information in the following way:

Variables: Variables fully describe the system. They can be discrete or continuous; scalar or vector.

Information: The information about N variables in the system is a probability density function (PDF) over the N-dimensional base space of this variables.

Relationships: Variables, in general, cannot take arbitrary values independent of each other. There are functional relationships that indicate the interdependence of this variables.

State: The system is at a particular time in a particular state. The set of all possible values that the state variables can take is the state space.

As shown in figure 2, the environment is modelled as a stochastic finite-state machine (FSM) with a finite number of states X, with inputs (actions A sent from the agent) and outputs (observations Y and rewards R sent to the agent). The agent is also modelled as a stochastic FSM with a finite number of states S, with inputs (observations/rewards sent from the environment) and outputs (actions sent to the environment).

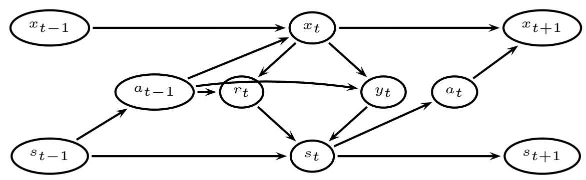

Figure 2: Agent environment interaction.

The state transitions of agent and environment satisfy the first order Markov property:

$$P(x_{t+1}|x_t, x_{t-1}, \ldots, x_0, a_t, a_{t-1}, \ldots, a_0) = P(x_{t+1}|x_t, a_t) \quad (3.1)$$

$$P(s_{t+1}|s_t, s_{t-1}, \ldots, s_0, r_{t+1}, r_t, \ldots, r_0, y_{t+1}, y_t, \ldots, y_0) = P(s_{t+1}|s_t, r_{t+1}, y_{t+1}) \quad (3.2)$$

where $x_{t-1}, \ldots, x_0$ and $s_{t-1}, \ldots, s_0$ are states from the past.

3.2 Information Processing

The information about a system is often not given in the appropriate form. Hence, information has to be processed in two different ways:

Transformation: The transformation involves very often data reduction. A N-dimensional PDF is transformed in a lower-dimensional PDF (or in a single number as in decision making).

Combination: This information processing type deals basically with the combination of new information with information that is already available. The tool for this task is the Bayes' rule.

There is one single purpose for an agent to spend time on information processing: to collect as much information as possible on the system, draw a conclusion or make a decision to perform an action. Information processing is performed every time the system transforms and/or obtains new information. At the level of information processing the concept of conflicting beliefs does not exist.

3.3 Decision Making

Decision making reduces the higher-dimensional PDFs to a single number or a low-dimensional vector of numbers.

A utility function is a mapping from the state space of the system to the real line. It represents the value of the information acquired thus far. The choice of the decision function is most often arbitrary. It is possible that the evaluation of two different utility functions may contradict each other.

Decisions can be taken at any time using information gathered up to the last moment. The decision does not transform the available information.

If we consider the dynamic system described by the state space model

$$x_{k+1} = f(x_k, u_k, \eta_k) \tag{3.3}$$

$$z_{k+1} = h(x_{k+1}, s_{k+1}, \xi_{k+1}) \tag{3.4}$$

where x is the system state vector, f and h nonlinear system and measurement function, z is the measurement vector, η and ξ system and measurement noise, u input vector of the state function, s for the sensor parameter vector as input of the measurement function, k the time step. The actions are the inputs to the system (states and measurements), denoted by $a_k = [u_k s_{k+1}]$. (inputs not given: active sensing).

To determine whether a policy (sequence of actions) $\pi_0 = [a_0 \dots a_{N-1}]$ is considered to result in a better performance than another policy a *performance criterion* (cost function) has to be chosen:

$$V^* = min_{\pi_0} V() = min_{\pi_0} \sum_j \alpha_j U_j(\dots) + \sum_l \beta_l C_l(\dots) \tag{3.5}$$

This criterion is a weighted sum of expected costs. The optimal policy π_0 is the one that minimizes this function. The j terms $\alpha_j U_j(\dots)$ characterizing the minimaization of the expected uncertainties $U_i(\dots)$ (maximization of the expected information extraction). The l terms $\beta_l C_l(\dots)$ denote other expected costs and utilities $C_l(\dots)$ such as time and energy. The choice of the weighting coefficients α_j and β_l reflect the preference is to collect information on the system or to minimize costs.

V is to be minimized with respect to the policy under certain constraints

$$c(x_0, \dots, x_N, \pi_0) \leq c_{max}. \tag{3.6}$$

The thresholds c_{max} express e.g. precedence constraint, machine constraint, order deadlines and so on.

4. MACRO-LEVEL INTERACTION

When the collection of agents becomes large the micro-modelling techniques described above becomes intractable, and other methods are called for. Specifically, in shop-floor control agents are required effectively to share manufacturing resources without knowing their exact identities, needs, or capabilities, and without there being a central authority that can, in a timely fashion, coordinate and manage this sharing. It is therefore natural to look for the mechanisms of choice in similar everyday situations, namely market mechanisms and stigmergy. Market mechanisms include the various price-seeking mechanisms ranging from simple auctions to highly complex exchanges. Similar in nature to financial exchanges and in particular to industry-specific exchanges, this will provide a forum for efficient allocation of resources among agents. Market mechanisms are decentralized and extremely adaptive and robust to changing conditions. However, they require a huge communication effort. Therefore, an indirect communication among agents is proposed to ensure that the individual decision making of the agents at the micro level is guided towards a global optimum. Here, agents place information in the environment rather than communicating mutually.

The aim of the macro level interaction is to identify the predicted posterior distribution of a global performance criterion (e.g. makespan) over the permutation of nodes in the JSP graph (policies). This is not an easy task considering the very large number of possible policies available. A sampling algorithm is called for where specially designed agents explore a solution, evaluate it and indicate the quality of it along the exploration path. Colorni et al. [4] suggested an approach based on ant colony optimization for job-shop scheduling. In this approach a number of homogenous non-communicative agents called "ants" travel along the JSP graph and explore a solution using a very simple heuristic (Figure 3). It includes a solution construction procedure in which an ant builds a solution of the JSP problem and a pheromone trail update procedure.

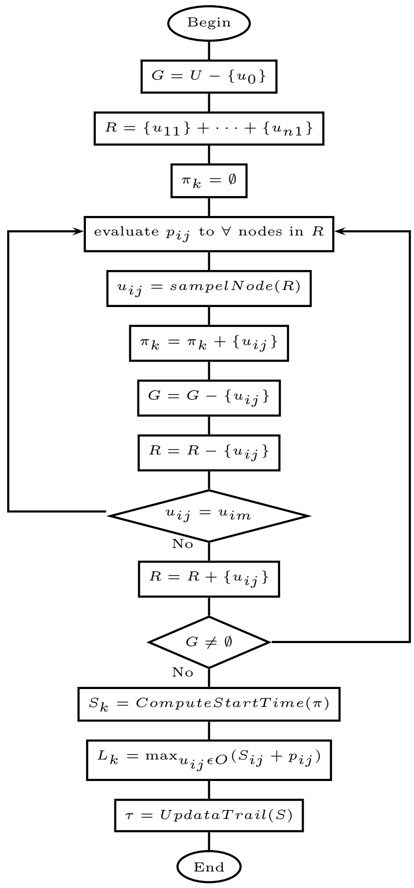

Figure 3: Ant heuristic.

The solution construction procedure is done probabilistically and can be represented in a state graph $Q = (U, E)$. The graph is composed of $N + 1$ nodes, with $U = O \cup \{u_0\}$ and $N(N-1)/2 + n$ edges with $E = \{(u_{ij}, u_{kl}) : u_{ij}, u_{kl} \varepsilon O\} \cup \{(u_0, u_{i1})\}$. for each solution. The nodes U represent $\pi(1, \dots, N)$ in equation 2.2.

Figure 4 presents an example of a state graph for the problem given in Figure 1. The shaded nodes represent explored states of the system, dots represent adjacent unexplored nodes. Solid lines between states represent state transitions. A solution is a chain of nodes adjacent to the node u_0. In this example there are three explored solutions:

$$\pi^1 = \{u_{11}, u_{31}, u_{12}, u_{13}, u_{21}, u_{32}, u_{33}, u_{22}, u_{14}, u_{23}, u_{34}, u_{24}\},$$
$$\pi^2 = \{u_{11}, u_{31}, u_{12}, u_{13}, u_{21}, u_{32}, u_{33}, u_{22}, u_{14}, u_{23}, u_{24}, u_{34}\},$$
$$\pi^3 = \{u_{11}, u_{31}, u_{12}, u_{13}, u_{21}, u_{32}, u_{14}, u_{22}, u_{23}, u_{33}, u_{34}, u_{24}\}.$$

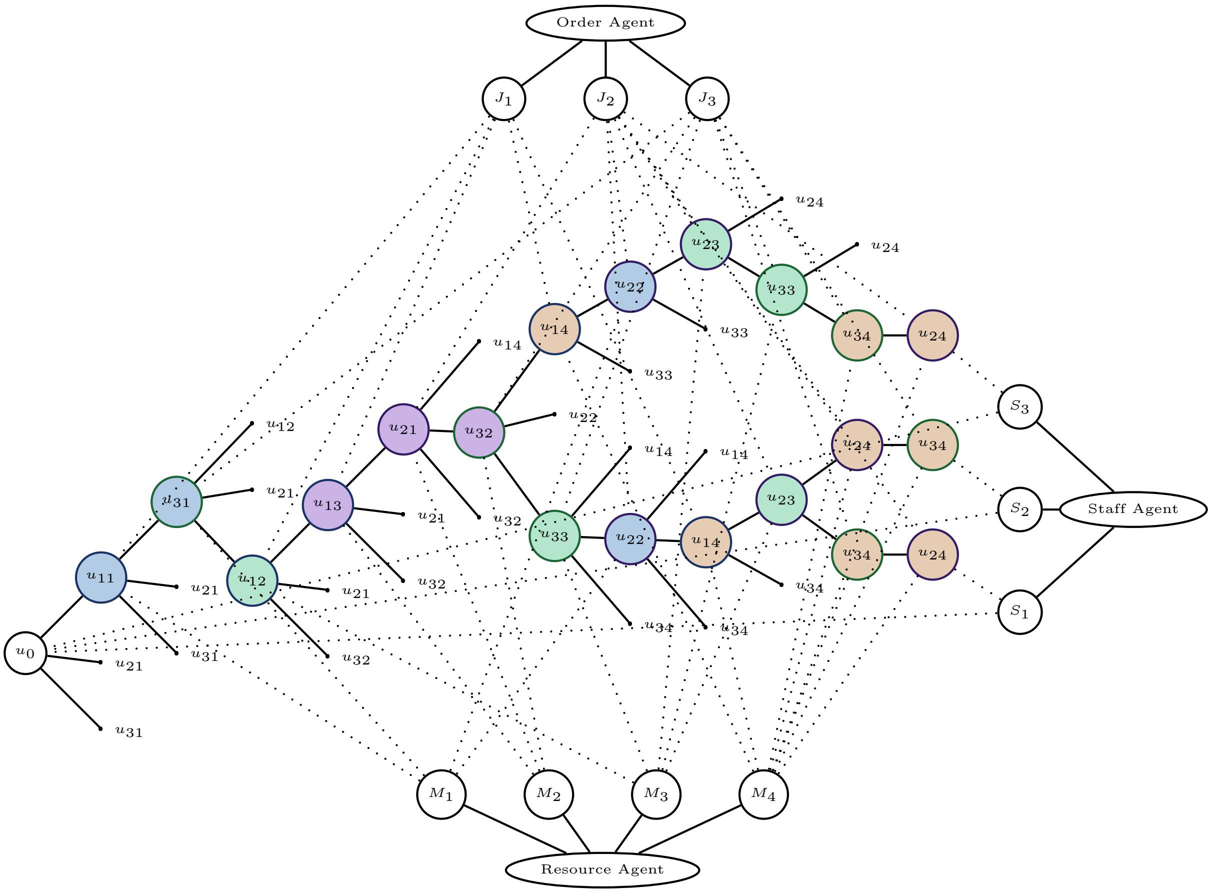

Figure 4: JSP state graph with agent interaction.

The probability with which each new node is added to the solution is a function of the item heuristic desirability η and the pheromone trail τ deposited by previous ants. Each arc in the JSP graph is weighted by $\{\tau_{ij}, \eta_{ij}\}$, where τ_{ij} is the *trail level* and η_{ij} the *desirability* of the transition $u(i) \to u(j)$.

$$p_{ij}^k(t) = \begin{cases} \frac{[\tau_{ij}(t)]^\alpha [\eta_{ij}]^\beta}{\sum_{j\varepsilon A_k} [\tau_{ij}(t)]^\alpha [\eta_{ij}]^\beta} & \text{if } j\varepsilon A_k : A_k = S \cap \pi, \\ 0 & \text{otherwise.} \end{cases} \tag{4.1}$$

A_k is the set of adjacent nodes from the JSP graph, which are not yet in the tabu list of the sampling circle.

The pheromone trail updating procedure changes the amount of pheromone trail in the edges of the graph. The trail matrix $T = [\tau_{ij}]$ memorizes structures in the search space that have been successfully exploited in the past. After each iteration k a $\triangle\tau_{ij}^k = \frac{Q}{L_k}$ is evaluated, quantifying the amount of trail left by iteration k on arc (ij), where Q is a system parameter. The total contribution of all iterations is:

$$\triangle\tau_{ij} = \sum_{k=1}^{m} \triangle\tau_{ij}^k \tag{4.2}$$

The trail levels to be used at the next iteration is given by

$$\tau_{ij}(l+1) = \rho\tau_{ij}(l) + \triangle\tau_{ij} \tag{4.3}$$

where ρ is the evaporation coefficient such that $(1-\rho)\tau_{ij}(l)$ represents the amount of trail which evaporates between iteration l and $l+1$. The evaporation coefficient can be influenced by agents using the effect of passive stigmergy, e.g. if they assume this information is no longer valid.

The visibility Matrix $H = [\eta_{ij}]$ is a problem specific greedy heuristic, giving other agents the opportunity to guide the exploration in a direction where they can expect to find a better solution.

5. COMBINATION OF MICRO- AND MACRO LEVEL

In addition to optimizing the individual goals of the agents, the macro-level implementation has to optimize a global objective corresponding to a performance criterion of the manufacturing system. These two objectives generate tradeoffs and possible conflicts. To regulate these a job of central command will eliminate such problems, steering the behavior of the units toward the global goal. To this end, central command will be viewed as a "market designer" that can set the reward mechanism in the system so as to ensure that by optimizing their own objectives the agents will also optimize the global objectives.

One problem with this approach has to do with the relative weakness of markets in resolving issues of pure coordination. When there are several possible joint actions for the agents that are optimal for everyone (which is the norm rather than the exception), the market mechanism does not automatically ensure that all agents will coordinate the same action. The approach of this paper is to investigate multi-agent learning, and have units in the field learn the optimal resource allocation strategy through a relatively brief learning phase. There has been considerable work in single agent learning. However, relatively little work in AI has been devoted to multi-agent learning, which calls for rather different techniques. Another problem has to do with having each agent compute its own utility function and, in particular, computing the value of information (VOI). While VOI is a well defined concept in decision theory, its computation is intractable.

An agent may notice that by trading in the market it is consistently supplying or consuming parts or information services of another agent and may decide to model that other agent explicitly. In a similar fashion, a group of agents may decide to form a manufacturing or an information team where there is agreement about tasks and interchange and thus interact with one another at the micro-level. Conversely, teams may dissolve when it is no longer beneficial to all members.

The homogenous non-communicative multi-agent system on the macro level uses two global data structures, the trail matrix and the visibility matrix. The ants or staff agent explore the state space. As indicated with a dotted line in figure 4 they travel from the current state of the system until a complete solution of the job shop is created and updates the trail matrix. The information in the trail matrix is not only used from the agents in the macro level, but also from agents in the micro level as a reward for their own actions. This allows a partial solution to be evaluated on a global scale. Agents in the micro level may also influence exploration at the macro level. They can use the visibility matrix to guide the ants in a direction where they expect better solutions from local point of view. Also, the actual search space, as it is indicated with a dotted line in figure 4, from a resource agent meeting in a state with a dotted line from a order agent, is created from interaction in the micro level. The interaction between micro and macro level is a heterogenous non-communicative multi-agent system.

6. SUMMARY

This paper presents a description of two interaction levels in multi-agent job-shop manufacturing systems, namely the micro-level and macro-level interaction. Micro-level interaction is the coordination of a small number of agents (order agent, resource agent and product agent) to solve a particular sub problem. This solution may involve modelling of other agents' decision behavior and reinforcement learning. A three phase strategy has been proposed: modelling, information processing and decision making. In the modelling phase a Bayesian network represents beliefs about factors that influence other agents' decision making processes. In the information processing phase, the agent collects, transforms and combines information. Finally, in the decision making phase, based on a utility function, the action is selected.

Macro-level interaction coordinates partial solutions from the micro-level to ensure that their decision making is synergistic rather than destructive. An ant system is used to identify the predicted posterior distribution of a global performance criterion over the permutation of possible partial solutions. The agents in the macro-level (ants) perform two procedures, the solution construction procedure and the pheromone trail updating procedure. In the solution construction procedure the ants assemble partial solutions to a global solution and evaluate their quality. In the pheromone trail updating procedure the quality of this solution is indicated along the path.

Micro-level and macro-level interact via two global data structures, the trail matrix and the visibility matrix. The trail matrix is generated by agents from the macro level and indicates to agents from the micro-level the quality of a partial solution. The visibility matrix is generated by agents from the micro level to guide the exploration process of agents at the macro-level.

7. ACKNOWLEDGMENTS

This work was supported in part by a GOA project grant of the Katholieke Universiteit Leuven, Department of Mechanical Engineering, Division PMA. Special thanks to Martin Kollingbaum, Bart Saint Germain and Margaret Mary Malone for comments on this manuscript.

8. REFERENCES

[1] R. Bajcsy. Active perception. In *Proceedings of the IEEE*, volume 79, pages 996–1005, 1988.

[2] O. Bochmann, P. Valckenaers, and H. V. Brussel. Negotiation-based manufacturing control in holonic manufacturing systems. In *Proceedings of the ASI'2000 Conference*, pages 206–214, Bordeaux, France, Sept. 2000.

[3] O. Bochmann, P. Valckenaers, and H. V. Brussel. Multi-agent manufacturing control with bayesian networks. In *Proceedings of the INCOM'01 Conference*, Vienna, Austria, Sept. 2001.

[4] A. Colorni, D. M., M. V, and T. M. Ant system for job-shop scheduling. *JORBEL–Belgian Jornal of Operation Research, Statistics and Computer Science*, 34:39–53, 1994.

[5] E. H. Durfee. Blissful ignorance: Knowing just enough to coordinate well. In *Proceedings of the First International Conference on Multi-Agent Systems*, pages 406–413, Menlo Park, California, 1995. AAAI Press.

[6] M. R. Garey, D. S. Johnson, and R. Sethi. The complexity of flowshop and jobshop scheduling. *Math. Oper. Res.*, 1:117–129, 1976.

[7] M. P. Georgeff and A. L. Lansky. Reactive reasoning and planning. In *Proceedings of the Sixth National Conference on Artificial Intelligence*, pages 677– 682, Seattle, WA, 1987.

[8] R. L. Graham, E. L. Lawler, J. K. Lenstra, and A. H. G. Rinnooy Kan. Optimization and approximation in deterministic sequencing and scheduling: A survey. *Annals of Discrete Mathematics*, 5:287–326, 1979. defines the JSP-Problem.

[9] P. P. Grassé. La reconstruction du nid et les coordinations inter-individuelles chez belicositermes natalensis et cubitermes sp. la théorie de la stigmergie : Essai d'interprtation du comportement des termites constructeurs. *Insectes Sociaux*, 6, 1959.

[10] C. Grossner, A. Preece, T. Radhakrishnan, and M. Newborn. Sharing data in multi-agent systems. *Submitted for review to International Journal of Cooperative Information Systems*, Nov. 1995.

[11] N. R. Jennings. Commitments and conventions: The foundation of coordination in multi-agent systems. *The Knowledge Engineering Review*, 8(3):223–250, 1993.

[12] A. S. Rao and M. P. Georgeff. BDI-agents: from theory to practice. In *Proceedings of the First Intl. Conference on Multiagent Systems*, San Francisco, 1995.

[13] B. Roy and B. Sussmann. Les problémes d'ordonnancement avec contraintes disjonctives. *Note DS*, 1964.

[14] P. Stone and V. Manuela. Multiagent systems: A survey from a machine learning perspective. *Autonomous Robotics*, 8(3):345–383, July 2000.

[15] M. Tambe. Recursive agent and agent-group tracking in real-time, dynamic environment. In *Proceedings of the First International Conference on Multi-Agent Systems*, pages 368–375, Menlo Park, California, 1995. AAAI Press.

[16] P. Valckenaers, H. V. Brussel, M. Kollingbaum, and O. Bochmann. Multi-agent coordination and control using stigmergy applied to manufacturing control. In M. Luck, V. Marík, O. Stepánková, and R. Trappl, editors, *EASSS*, volume 2086 of *Lecture Notes in Computer Science*, pages 317–334. Springer, 2001.

[17] H. Van Brussel, J. Wyns, P. Valckenaers, L. Bongaerts, and P. Peeters. Reference architecture for holonic manufacturing systems. *Computers in Industry, special issue on IMS*, 37(3):255–276, 1998.