
Autonomous Agents and Multi-Agent Sytems, 8, 237–266, 2004

Wasp-like Agents for Distributed Factory Coordination

VINCENT A. CICIRELLO cicirello@cs.drexel.edu
Department of Computer Science, Drexel University, 3141 Chestnut Street, Philadelphia, PA 19104, USA

STEPHEN F. SMITH sfs@cs.cmu.edu
The Robotics Institute, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213, USA

Abstract. Agent-based approaches to manufacturing scheduling and control have gained increasing attention in recent years. Such approaches are attractive because they offer increased robustness against the unpredictability of factory operations. But the specification of local coordination policies that give rise to efficient global performance and effectively adapt to changing circumstances remains an interesting challenge. In this paper, we present a new approach to this coordination problem, drawing on various aspects of a computational model of how wasp colonies coordinate individual activities and allocate tasks to meet the collective needs of the nest.

We focus specifically on the problem of configuring parallel multi-purpose machines in a factory to best satisfy product demands over time. Wasp-like computational agents that we call routing wasps act as overall machine proxies. These agents use a model of wasp task allocation behavior, coupled with a model of wasp dominance hierarchy formation, to determine which new jobs should be accepted into the machine's queue. If you view our system from a market-oriented perspective, the policies that the routing wasps independently adapt for their respective machines can be likened to policies for deciding when to bid and when not to bid for arriving jobs.

We benchmark the performance of our system on the real-world problem of assigning trucks to paint booths in a simulated vehicle paintshop. The objective of this problem is to minimize the number of paint color changes accrued by the system, assuming no a priori knowledge of the color sequence or color distribution of trucks arriving in the system. We demonstrate that our system outperforms the bidding mechanism originally implemented for the problem as well as another related adaptive bidding mechanism.

Keywords: distributed scheduling, dynamic scheduling, biologically inspired system, factory coordination.

1. Introduction

Effective coordination of multiple agents interacting in dynamic environments is an important part of many real-world problems. For example, teams of robots exploring alien terrain need to coordinate such activities as task assignment and scheduling among the team members; groups of agents comprising web-based information tools need to coordinate such activities as information gathering, sharing, and so forth. In such dynamic domains, effective global behavior is sometimes best achieved by localized, adaptive coordination policies.

Our particular interest in this paper is the domain of factory operations. The factory is a complex dynamic environment and manufacturing organizations are constantly faced with the need to rearrange production. New and evolving market opportunities lead to changing product demands and manufacturing priorities. Changes in resource availability affect production capacity and force reassessment of

current production goals. Such changing circumstances are quite frequently at odds with common elements of factory operations – for example, production scheduling and attempts to build schedules in advance. Though advance scheduling can provide a basis for configuring factory resources to optimize performance relative to (currently) known requirements and constraints, these prescriptive solutions also tend to be quite brittle and they can quickly become invalidated by unexpected events.

In practice, manufacturing operations are often coordinated in a decentralized manner. The use of local dispatch scheduling policies, for example, is commonplace in many manufacturing environments [43]. By making decisions only when needed to keep execution going and by basing them on aspects of the current dynamic state, dispatch-based strategies can be quite insensitive to unexpected events and yield very robust behavior. This advantage can also be a disadvantage, however, as decisions are made myopically and this can lead to sub-optimal factory performance.

The desire for a more robust basis for coordination has also motivated research into agent-based approaches to manufacturing scheduling and control (e.g., [10, 38, 39, 45, 46, 51]) and there have been a few interesting successes. For example, Morley uses a simple bidding mechanism for assigning trucks to paint booths at a General Motors factory [41, 42]. They have shown that their decentralized approach, which imparts decision-making ability upon the paint booths themselves, significantly outperformed the previous centralized scheduling system in terms of increased throughput and lower paint costs.

However, decentralized approaches can sometimes also be susceptible to sub-optimal and even chaotic global behavior. Kempf and Beaumariage show how a distributed manufacturing system, utilizing extremely simple dispatch policies, can exhibit formally chaotic behavior [2, 35]. Their simple example system has a number of attractors with drastically different global performance. They show that small changes in policies or in system state can force the system to move between these attractors leading to large changes on a global scale. In general, the ability to orchestrate good global performance via local interaction protocols and strategies remains a significant and ill-understood challenge.

One approach to this class of problem is to view establishment of appropriate coordination policies as an adaptive process (i.e., policies that adapt to dynamically changing circumstances). There are many examples of effective, adaptive behavior in natural multi-agent systems (e.g., [29, 34, 36, 54]), and computational analogies of these systems have served as inspiration for multi-agent optimization and control algorithms in a variety of domains and contexts (e.g., 3, 5, 24, 48]). Bonabeau et al. provide a comprehensive survey of adaptive multi-agent systems that have been inspired by social insect behavior [4].[1]

In this paper, we present a system for dynamically scheduling parallel multi-purpose machines based on the natural multi-agent system of the wasp colony. A computational model of real wasp behavior [7, 52–54] lies at the foundation of our approach. In this model, interactions between individual wasps and the local environment (wasp-to-environment interactions) in the form of a stimulus-response mechanism govern distributed task allocation. Furthermore, interactions between pairs of individual wasps (wasp-to-wasp interactions) result in the self-organization of dominance hierarchies. We combine these two aspects of wasp behavior into an

effective, decentralized basis for coordinating the assignment of jobs to machines in a factory. Our approach is shown to be competitive to the "real-world proven" market-based truck painting system of Morley in which resources (e.g., vehicle paint booths) bid for jobs (trucks) as they arrive [41, 42].

Our approach is also shown to be superior to a wasp-inspired bidding model proposed by the creators of the underlying wasp behavior model [11]. In Campos et al.'s system, the bidding agents associated with the resources use an adaptive bid formulation based on the same wasp model that we employ, but in a manner quite different. They use the model to adjust bid determination rules; while our wasp-based bidding agents adjust decision policies of whether to bid or to not bid based on job type. Morley's bid strategy as well as the adaptive policy of Campos et al. both require a machine to bid if there was space in its queue; whereas our adaptive policy allows non-bids to occur in anticipation of near-future jobs of a type for which the machine is better-suited. It is the combination of the adaptive nature of the policy and the ability to choose not to bid that gives our approach its strength.

The remainder of the paper will proceed as follows. In Section 2 we present our wasp-inspired adaptive bidding mechanism for the dynamic assignment of jobs to parallel multi-purpose machines. Section 3 describes the vehicle paintshop problem that is the focus of our experimentation. It further details the system of Morley and the system of Campos et al. Section 4 benchmarks our system to the bidding agents of Morley and Campos et al. Section 5 analyzes the behavior of the wasp-like agents through an examination of the response thresholds adaptation over time. We discuss limitations of our approach in Section 6. We provide a summary of related work in Section 7 and conclude in Section 8.

2. Our approach: R-Wasps

In this section, we present our approach to the problem of allocating dynamically arriving jobs to machines in a factory setting. The general problem that we are interested in is comprised of some number of parallel machines. Each machine is multi-purpose and is capable of processing some number of job types, but with a cost to reconfigure the machine from one type to another. It is therefore beneficial to minimize such setups in order to maximize system output. If the incoming flow of new jobs allows, then ideally each of the machines should specialize to one or a few types of jobs from among the ones it is capable of processing. Our system adapts such machine specializations via a mechanism that has been inspired by the adaptive task allocation behavior of real wasps. In Section 2.1 we summarize the underlying model of wasp behavior. Then in Sections 2.2 and 2.3 we give a detailed formulation of our wasp-inspired system that we call *R-Wasps*.

2.1. Wasp behavior model

Theraulaz et al. present a model for the self-organization that takes place within a colony of wasps [54].

The model of wasp behavior describes the nature of interactions between an individual wasp and its local environment with respect to task allocation [54]. They model the colony's self-organized allocation of tasks using what they refer to as response thresholds. An individual wasp has a response threshold for each zone of the nest. Based on a wasp's threshold for a given zone and the amount of stimulus from brood located in that zone, a wasp may or may not become engaged in the task of foraging for that zone. A lower response threshold for a given zone amounts to a higher likelihood of engaging in activity given a stimulus. Bonabeau, et al. [7] discuss a model in which these thresholds remain fixed over time. Later they considered that a threshold for a given task decreases during time periods when that task is performed and increases otherwise [53]. Bonabeau et al. [6] demonstrate how this model leads to a distributed system for allocating mail retrieval tasks to a group of mail carriers. Although they deal with a "toy" problem, they illustrate the potential of systems inspired by the underlying wasp behavioral model. Campos et al. [11] take the model a step further illustrating a connection between market-based mechanisms and wasp-inspired systems. They apply a system that is loosely inspired by their natural model to a simulation of the real-world vehicle paintshop problem of Morley. The insect-inspired bidding rules of Campos et al. result in a slight improvement over the market-mechanism of Morley.

The system that we present in this paper is likewise inspired by this wasp behavior model. But we incorporate aspects of the model which have been ignored by Campos et al. Specifically, as will be discussed later, Campos et al.'s system equates low response thresholds to high bids and high response thresholds to low bids and models their bid determination rule to incorporate these thresholds. But in the actual model of wasp behavior, response thresholds signify a pre-disposition to respond to a stimulus rather than specifying the degree of response. Considering this, our system uses the wasp model and the response thresholds to formulate an adaptive policy to decide whether or not to bid at all for a job.

The model of wasp behavior also describes the nature of wasp-to-wasp interactions that take place within the nest [54]. When two individuals of the colony encounter each other, they may with some probability interact with each other in a dominance contest. If this interaction takes place, then the wasp with the higher social rank will have a higher probability of dominating in the interaction. Through such interactions as these, wasps within the colony self-organize themselves into a dominance hierarchy. Theraulaz et al. [52] discuss a number of ways of modeling the probability of interaction during an encounter which range from always interacting to interacting based upon certain tendencies of the individuals. We further incorporate this aspect of the behavior model into our system to determine the winning bidder. That is, when two or more of our wasp-like agents bid for a given job, the winner is chosen through a tournament of dominance contests.

2.2. *Routing wasps*

Each multi-purpose machine in our system has an associated *routing wasp*. Each routing wasp is in charge of choosing which jobs to bid on for possible assignment to

the queue of its associated machine. Each routing wasp has a set of response thresholds, much like that of the underlying wasp behavior model:

$$\Theta_w = \{\theta_{w,0}, \ldots, \theta_{w,j}\} \tag{1}$$

where $\theta_{w,j}$ is the response threshold of wasp w to jobs of type j.

Jobs in the system that have not yet been assigned to a machine and that are awaiting assignment broadcast to all of the routing wasps a stimulus S_j which is proportional to the length of time the job has been waiting for assignment to a booth. This stimulus is "typed" according to the type of job. So the longer the job remains unassigned, the stronger the stimulus that it emits. A routing wasp w will bid for a job that is emitting a stimulus S_j with probability:

$$P(\text{bid}|\theta_{w,j}, S_j) = \frac{S_j^2}{S_j^2 + \theta_{w,j}^2} \tag{2}$$

Otherwise it will choose not to bid. This is the rule defined for task allocation in the wasp behavioral model as described in [53]. The exponent of "2" can be seen as a system parameter. This is the value used in the original wasp model and it appears to work well experimentally in our domain. In this way, wasps will tend to bid for jobs of the type for which its response threshold is lowest. But they will bid for jobs of other types if a high enough stimulus is emitted. This mechanism for deciding whether or not to bid for a job is illustrated in Figure 1.

The threshold values $\theta_{w,j}$ may vary in the interval $[\theta_{\min}, \theta_{\max}]$. Each routing wasp, at all times, knows what its machine is doing, including: the status of the queue, whether or not the machine is performing a setup, the type of job it is processing, and whether or not the machine is idle. This knowledge is used to adjust the response thresholds for the various job types. This updating of the response thresholds occurs at each time step. If the machine is currently processing a job of type j or is in the process of setting up to process type j, then $\theta_{w,j}$ is updated according to:

$$\theta_{w,j} = \theta_{w,j} - \delta_1 \tag{3}$$

If the machine is either processing or setting up to process a job type other than j, then $\theta_{w,j}$ is updated according to:

$$\theta_{w,j} = \theta_{w,j} + \delta_2 \tag{4}$$

And if the machine is currently idle and has an empty queue, then for all job types j the wasp adjusts the response thresholds $\theta_{w,j}$ according to (t is the length of time the machine has been idle and is an exponent):

$$\theta_{w,j} = \theta_{w,j} - \delta_3^t \tag{5}$$

The δ_1, δ_2, and δ_3 are positive system constants. Consequently, the response thresholds for the job type currently being processed are reinforced as to encourage

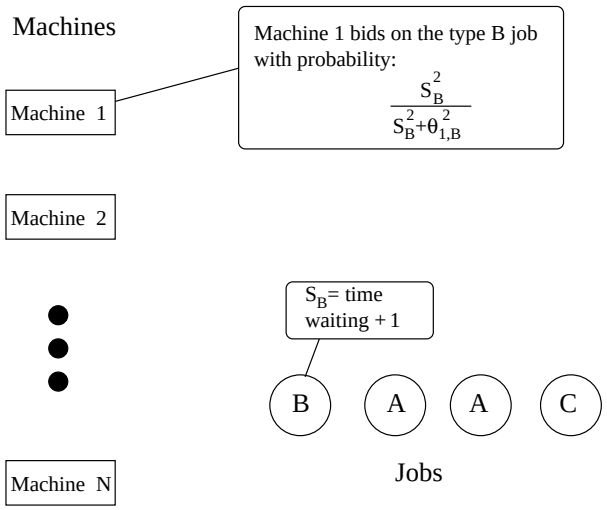

Figure 1. Each machine is represented by a wasp-like agent called a routing wasp. This routing wasp maintains response thresholds $\theta_{w,i}$ for each job type i. Given a stimulus for a job of type i, the routing wasp stochastically decides whether or not to bid for the job according to the type of job, the length of time the job has been waiting, and the response threshold for that type.

the routing wasp to bid on jobs of the same type; while the response thresholds of other job types not currently being processed are adapted to discourage the routing wasp from bidding on these job types. This specialization of routing wasps (i.e., machines) helps to minimize setup time. The first two ways in which the response thresholds are updated (Equations (3) and (4)) are analogous to that of the real wasp model [6, 53]. The third (Equation (5)) is included to encourage a wasp associated with an idle machine to take whatever jobs it can get rather than remaining idle. This last update rule acknowledges that although specialization can reduce setup time, over-specialization to a job type with low demand may result in lower system throughput.

2.3. Dominance contests

The routing wasp formulation of the previous section does not state what happens if two or more routing wasps respond positively to the same stimulus (i.e., two machines bid on a job). One simple approach is to make the decision randomly [19]. But there is a problem with this. Consider the case where one machine has been sitting idle and has an empty queue. Perhaps this machine has specialized to a job type whose demand has diminished. Now consider a second machine with a long queue of jobs. Suppose a new job arrives at the factory of the type for which this second machine has developed a preference. The idle machine by this point is willing to take any job. Both machines respond to the stimulus from this new job. In the naive random decision, both machines would have an equal probability of taking on this new job. But perhaps the idle machine with the empty queue should have a higher

probability of getting the job even though it is not currently configured for this job type and will accrue some setup time.

To this end we augment the basic routing wasp model with a method for deciding which routing wasp from a group of competing wasps gets the job. This method is based on the self-organized social hierarchies of real wasps (see [52, 54]). First define the force F_w of a routing wasp w as:

$$F_w = 1.0 + T_p + T_s \tag{6}$$

where T_p and T_s are the sum of the process times and setup times of all jobs currently in the queue of the associated machine, respectively.[2] The values of T_p and T_s are easily computed given the queue of jobs. Now consider a dominance struggle between two competing routing wasps. This contest determines which routing wasp gets the job. Let F_1 and F_2 be the force variables of routing wasps 1 and 2, respectively. Routing wasp 1 will get the job with probability:

$$P(\text{Wasp 1 wins}|F_1, F_2) = \frac{F_2^2}{F_1^2 + F_2^2} \tag{7}$$

In this way, routing wasps associated with machines of equivalent queue lengths will have equal probabilities of getting the job. If the queue lengths differ, then the routing wasp with the smaller queue has a higher probability of taking on the new job.

In the event that more than two routing wasps compete for a given job, a single elimination tournament of dominance contests is used to decide the winner. Seedings in this tournament are according to the values of the force variables of the competing wasps. To deal with odd numbers of competing wasps, the $2^{\lceil \log_2 (C) \rceil} - C$ wasps with highest values of F_w, where C is the number of competing wasps, receive "buys" to the second round (i.e., they do not have to compete in the first round). An example of such a dominance tournament can be seen in Figure 2.

3. The problem: Vehicle paintshop booth assignment

The problem that we focus on to evaluate our approach is that of dynamically assigning trucks to paint booths in a simulated vehicle paintshop. Morley describes the paintshop of a specific General Motors plant [41, 42]. Many details of the actual factory were changed in Morley's simulation to protect company secrets. But the problem is interesting just the same. In Morley's problem, trucks roll off the assembly line at a rate of one per minute. The system is faced with the problem of dynamically assigning each truck to a paint booth as it emerges from the end of the assembly line. If the queues of all paint booths are full, then the truck waits in a "storage" location while trucks continue to arrive from the assembly line. There are seven paint booths in Morley's problem and the queue of each can contain at most three trucks due to physical space constraints within the plant. It takes 3 minutes to

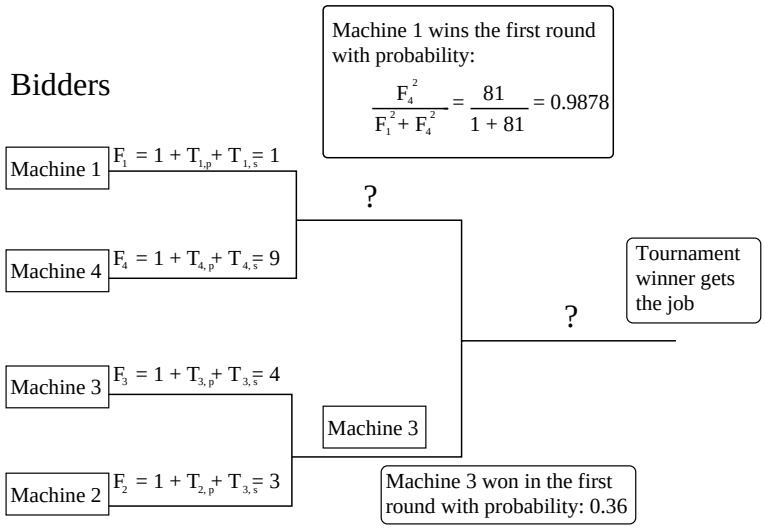

Figure 2. All those routing wasps (machines) that decided to bid for a job compete in a tournament of dominance contests. Seedings in the tournament are according to their force variables F_i. The job is assigned to the queue of the winner.

paint a truck. Each truck can possibly require any of 14 paint colors and the trucks arrive in no particular order (i.e., the required colors of future trucks are not known until they arrive from the assembly line). Approximately 50% of the trucks require a single color. The other 50% require colors drawn uniformly at random from among the other 13 colors. A paint booth can only be set for one color at a time and there is a cost to reconfigure the booth for another color in terms of both the time it requires to perform this color change as well as a monetary cost associated with paint usage. The time cost is 1 minute (we also consider in our experiments later in Section 4 a variation with a much more significant time cost of 10 minutes). But it is the latter monetary cost that is most important. During a color change, some amount of paint is wasted when the system is flushed of the previous paint color. Morley did not specify the quantity of wasted paint in his description of the problem, but according to Braslaw [8] of Ford Research, approximately $\frac{1}{6}$–$\frac{1}{8}$ of the paint required by the base coat and primer to paint a vehicle is wasted during a color change in a typical paintshop. In any event, this should serve to illustrate that a control policy that assigns trucks to paint booths as to minimize such color changes is very desirable. Achieving this objective is a difficult challenge due to lack of knowledge of the colors of future arriving trucks as well as other uncertain events such as paint booth failures and so forth.

3.1. Bidding agents

Morley devised a simple bidding mechanism in which booth agents submit bids for trucks as they arrive according to their current queue length and the required color

of the last truck in the queue. This simple multi-agent bidding system was shown in simulation to be more effective than the previously used centralized scheduler, and was subsequently put into use at a General Motors truck painting facility. When put into practice in the GM facility, Morley's system was found to be 10% more efficient (in terms of number of paint color changes) than the previously used centralized scheduler [42] and resulted in savings of nearly a million dollars in the first 9 months of use [41].

Although Morley's system is formulated and implemented as a simple bidding mechanism (following optimization of the parameters of the bid determination rule via a genetic algorithm) his system simplifies to the following three rules considered in this order (as described in [42]):

1. Assign the truck to the booth with the shortest queue (with space) whose last truck is of the same color as this next truck (if such a booth exists).
2. Assign the truck to a booth with an empty queue if such a booth exists.
3. Assign the truck to a booth with the shortest queue (with space) if such a booth exists.

Rule 2 is redundant and is actually encapsulated in rule 3 but is included here to remain consistent to Morley and Schelberg [42].

3.2. *Adaptive bids*

Campos et al. [11] define an adaptive bid determination rule inspired by the self-organized task allocation of wasps. Their bid determination rule actually differs quite a bit from the model of wasp behavior though it is clearly inspired by it. They use the model to define a formula for the bids made by their agents and each booth agent bids as long as there is space in its queue. In R-Wasps alternatively we use the model to define a policy for whether or not to bid at all. The rule used by Campos et al.'s agents is as follows:

$$B_k = \frac{D_c^2}{D_c^2 + \alpha \cdot \theta_{k,c}^2 + \Delta T_k^{2\cdot\beta}} \tag{8}$$

The booth k with the highest value of this B_k is assigned the truck. α and β are system parameters. ΔT_k is the sum of the processing times and any necessary setup times of all trucks in the queue of booth k plus any remaining processing time of the truck currently being painted by the booth if such a truck exists. $\theta_{k,c}$ is the *response threshold* of booth k to trucks of color c. The concept of a response threshold comes from the model of wasp task allocation behavior as we have discussed earlier. Only here, response thresholds vary somewhat from their meaning in the wasp behavior model. In the wasp behavior model, as well as in its application within R-Wasps, response thresholds specify pre-dispositions to respond to stimuli; while in Campos et al.'s system response thresholds instead specify magnitudes of response. D_c is a

calculation of the demand for color c among trucks in the system that have not yet been assigned to paint booths (i.e., the new truck and any trucks in the "storage" location).

One thing to note in this definition is that the value of D_c is independent of the paint booth agent that is bidding on the truck. It depends only on the currently unallocated trucks. Due to this and since Campos et al.'s system requires all booth agents to bid and the truck is assigned to the paint booth k with the highest value of B_k, the above bid rule B_k can be simplified. The booth independence of D_c allows us to treat it as a constant. It is a completely superfluous term in the definition of B_k. The actual value of D_c is irrelevant in determining the booth k with the maximum B_k and choosing the maximum B_k in the above rule is equivalent to choosing the maximum B_k in the following rule:

$$B_k = \frac{1}{1 + \alpha \cdot \theta_{k,c}^2 + \Delta T_k^{2 \cdot \beta}} \tag{9}$$

The response thresholds $\theta_{k,c}$ of Campos et al.'s rule adapt in a manner analogous to that of the computational model of wasp behavior, but somewhat differently as compared to R-Wasps. Just as in the model of real wasp behavior, the response thresholds of Campos et al. adapt according to the activity of the paint booths. The difference between Campos et al.'s approach and R-Wasps is that Campos et al. updates the response thresholds only when a truck is assigned to a paintbooth; whereas we update the response thresholds on an ongoing basis at discrete time intervals according to booth activity. In Campos et al.'s system, when a truck of color c is assigned to booth k, the value of $\theta_{k,c}$ is decreased as follows:

$$\theta_{k,c} \leftarrow \theta_{k,c} - \xi \tag{10}$$

Furthermore, the values of $\theta_{m,c}$ for all other booths m not equal to k are increased according to

$$\theta_{m,c} \leftarrow \theta_{m,c} + \phi \tag{11}$$

The ξ and ϕ are system parameters. The $\theta_{k,c}$ are allowed to vary in the interval $\theta_{\min}$ to $\theta_{\max}$. These update rules are applied once each time a truck is assigned to a booth. In R-Wasps, alternatively the update rules occur at discrete time intervals. Also recall that R-Wasps incorporates a third idle machine update rule; whereas Campos et al. just include the two rules directly inspired by the wasps.

In a spirit much like that of Morley, Campos et al. use a genetic algorithm at the metalevel to optimize the system parameters for Morley's domain. The resulting parameter values are: $\alpha = 617.188$, $\beta = 4.66797$, $\xi = 7.85156$, $\phi = 17.7344$, $\theta_{\min} = 5.50781$, and $\theta_{\max} = 39.6875$.

In our own experimentation, we discovered that this set of parameter values does not lead to very good results for Morley's domain. In particular, the parameter set specified by Campos et al. does not perform well for the primary objective criteria

specified by Morley's problem – minimizing the number of paint color changes. It does, however, appear to do well for a somewhat related objective of minimizing average cycle time as we will see in the experiments in later sections. To understand why, compare the bid for a truck of color c made by a booth specialized to color c (i.e., response threshold for c is $\theta_{\min}$) to the bid of a booth highly unspecialized to color c (i.e., response threshold for c is $\theta_{\max}$). Assume further that the unspecialized booth has a ΔT 1 minute less than that of the specialized booth (i.e., the smallest difference between the ΔT values allowed by the problem). Now, let the ΔT of the unspecialized booth be equal to or greater than 4 minutes; and let the ΔT of the specialized booth be 1 minute greater than that. Using the set of parameters that Campos et al. indicate, the truck would be assigned to a booth that would have to change colors over one that would not have to change colors if it would save a single minute of cycle time. To be fair to Campos et al., they emphasize that they are merely giving an example of how the insect-inspired approach can be of use to real-world problems, and not necessarily tuning performance to the problem's true objective.

Given this, we optimized the parameters of their system ourselves using a simple hand-tuning technique.[3] In the experiments reported below in Section 4, we use both their set of parameters as well as the hand-tuned set. The hand-tuned parameter set is the following: $\alpha = 1.3$, $\beta = 1.5$, $\xi = 8.0$, $\phi = 16.0$, $\theta_{\min} = 0.0$, and $\theta_{\max} = 50.0$.

4. Experiments

4.1. Experimental design

In this Section, we detail a comparison between our system, Morley's system, and Campos et al.'s system on the real-world problem summarized earlier for which Morley's system was originally designed and implemented. Results of our system are indicated by "R-Wasps" and results of Morley's system as "Morley". Two configurations of Campos et al.'s system are considered: (1) using the original parameter set (Campos1); and (2) using the new parameter set (Campos2).

Our implementation of this problem has seven paint booths and 14 paint colors. Each paint booth is initially configured for a paint color chosen uniformly at random and has a queue limit of three trucks. Trucks arrive at a rate of one every simulated minute. The painting time of a truck is equal to three simulated minutes.

In what follows, we consider a number of problem variations that vary the distribution of paint colors of arriving trucks (including a dynamically changing color distribution), the amount of setup time required to change the paint color, and the probability of machine breakdowns:

- Problem 1: Morley's original problem
- Problem 2: Very significant setup times (heavily loaded system)
- Problem 3: Probability of machine breakdown
- Problem 4: Higher probability of machine breakdown
- Problem 5: Alternative color distribution
- Problem 6: Dynamically changing color distribution

We consider each of these problems separately below. In each experiment, the entire simulation runs for 1000 simulated minutes. The simulation however advances in time steps of 12 simulated seconds. In R-Wasps, the response threshold updates occur every time step (five times per simulated minute) according to the activities of the machines during that time step. In the system of Campos et al., the response threshold updates only occur upon assignment of a job to a machine as specified by their system formulation.

To handle the queue length limit of three trucks, we add a new condition to the rule used by the routing wasps in determining whether or not to respond to a stimulus from a truck. If a queue is full (contains three trucks), the routing wasp in charge of that queue does not respond to any truck stimulus. Otherwise, it uses the stochastic rule described in Section 2. The system parameters for R-Wasps for this problem have been set as follows via a simple hand-tuning procedure: $\theta_{\min} = 1$, $\theta_{\max} = 10,000$, $\delta_1 = 100$, $\delta_2 = 10$, and $\delta_3 = 1.05$. Furthermore, the initial values of the $\theta_{w,j}$ are set randomly but correlated to the initial paint color c of the paint booth w. If $c = j$, then the initial value for $\theta_{w,j}$ is $\theta_{\min}$. Otherwise, $\theta_{w,j}$ is selected uniformly at random from the interval $\left[\frac{\theta_{\max}}{2}, \theta_{\max}\right]$. Also, to handle the finite-sized "storage location" of Morley's problem, booths are required to bid if the storage location is full (provided there is space in their queue). This requirement actually turns out to be unnecessary for the experiments that follow.

To compare the performance of R-Wasps, Morley, Campos1, and Campos2, we measure the average number of setups accrued by each of these systems across 100 independent runs of each of the six described problem variations. The number of setups is directly related to the amount of wasted paint and is the primary objective of Morley's problem. We also measure the average cycle time,[4] average throughput,[5] and the average queue length[6] of each of these systems on the variations of Morley's problem.

4.2. *Morley's original problem*

Let us begin the experimental comparison by looking to the results for Morley's original problem (Problem 1) which can be found in Table 1. This is Morley's originally described problem. Setup time to change between paint colors is 1 minute. The color distribution is:

- With probability 0.5, a truck is of color C1.
- A truck has a probability of $\frac{1}{26}$ of being color C2 (similarly for colors C3, . . ., C14).

The results are averages of 100 independent runs of the simulation and show both average cycle times as well as average number of setups (color changes).

In terms of the number of setups performed on average, R-Wasps performs significantly fewer setups than all of the other three systems. As stated earlier, the principal objective in this problem is minimizing the number of setups. Under the assumption that the number of setups is indicative of the extra cost associated with additional paint usage, it is clear from the comparison of the number of setups that

Table 1. Problem 1: Setup time is less significant in this problem.

	R-Wasps	Morley	Campos1	Campos2
Num. setups	287.61 ± 2.15	438.22 ± 3.70	530.52 ± 4.05	369.38 ± 4.07
Cycle time	7.16 ± 0.10	3.87 ± 0.03	3.53 ± 0.001	6.65 ± 0.03
Throughput	994.03 ± 0.36	997.09 ± 0.21	997.53 ± 0.10	994.43 ± 0.19
Queue length	0.05 ± 0.02	0.00 ± 0.00	0.00 ± 0.00	0.01 ± 0.01

95% confidence intervals are shown.

R-Wasps accrues significantly less paint costs. In fact, assuming a linear relationship between number of setups and such paint costs, R-Wasps shows a savings of 34% over Morley's system, a savings of 46% over Campos1, and a savings of 22% over Campos2. You can also note that in terms of cycle time, Morley's and both versions of Campos et al.'s system perform better than R-Wasps. Campos1 (i.e., with Campos et al.'s original system parameters) actually has the lowest average cycle time.

A natural question to ask at this point is "why with less setups we are not seeing smaller average cycle times?" There are a couple of reasons for this behavior. The first is that R-Wasps through its response threshold adaptation is able to specialize a couple of booths to the more highly demanded color, so colors of lesser demand will tend to be assigned to the queues of other booths not specialized to this high demand color even if these queues are longer. A second reason is that through the ability of machines to "not bid," R-Wasps is able to put off assignment of arriving trucks for a few time units to allow for better sequencing (from the standpoint of number of setups). In Morley's system and in Campos et al.'s system, such delayed job assignment is not considered (i.e., all machines bid provided there is space in their queues); while R-Wasps uses a stochastic rule to choose whether or not to bid at all even in cases where there is space in the queue. This ability to not bid can allow a paint booth with a longer queue, but which would not require a setup, to be assigned an arriving truck over a paint booth with a very short or empty queue but which would require a setup. When all booths bid for all trucks (as in both Morley and Campos), this may not be the case and the booth requiring the setup might be assigned the truck. There is a tradeoff going on in the problem between minimizing setups and minimizing cycle time. The solution that optimizes one does not necessarily optimize the other. In Problem 1, the system can paint seven trucks every 4 minutes on average in the worst case where every truck requires a paint flush, while only four new trucks are arriving during that 4 minutes. With approximately 43.8% of its trucks requiring a paint flush, Morley's system is much closer to this worst case than is R-Wasps. Campos1 is even worse with 53.1% of its trucks requiring a paint flush, although Campos2 is a bit better with 36.9% of its trucks requiring setup. But R-Wasps, utilizes this difference in production capacity and truck arrival rates; and through paint booth specialization and delayed job assignments, R-Wasps is better able to sequence the trucks in queues as to minimize number of setups with only approximately 28.8% of jobs requiring setup.

4.3. *More significant setup times*

Let us next turn to the results of the more heavily loaded system with much more significant setup times (Problem 2) which can be found in Table 2. In this Problem 2, setup time to switch between paint colors is increased from 1 to 10 minutes. All other problem aspects remain the same as in Problem 1. The results are averages of 100 independent runs of the simulation and show both average cycle times as well as average number of setups (color changes).

First note that both versions of Campos et al.'s system perform very poorly both in terms of average cycle time and number of setups as compared to Morley's original system as well as R-Wasps. The reason for Campos et al.'s poor system performance is that a paint booth in their system considers the sum of painting times and flush times for all trucks in its queue in its bid formulation. The problem is that both the parameters tuned by Campos et al. (Campos1) and those tuned by us (Campos2) have been tuned for Morley's instance (Problem 1) which has a much less significant paint flush time. The response threshold term of their formulation ends up dominated by the ΔT term in Problem 2. R-Wasps was also tuned for Problem 1, but the control parameters of R-Wasps appear to be far less sensitive to problem characteristics than the parameters of Campos et al.'s system. R-Wasps' decision of whether or not to bid is based solely on the color of the truck, how long that truck has been waiting, and the response threshold to that color. It does not consider paint times nor flush times. Instead, our response threshold updating rules are applied at every time step and updated appropriately according to the color the booth is either painting or setting up to paint. So the longer the setup time in a problem instance, the stronger the response threshold updates will be in the R-Wasps system. In a sense, our formulation self-tunes its sensitivity to paint and flush times; while Campos et al.'s system requires tuning a parameter related to that sensitivity beforehand assuming you know enough about the problem to do so.

Next, note that R-Wasps significantly outperforms Morley's system for Problem 2 both in terms of average cycle times as well as average number of setups. This performance difference is likely due to the routing wasps ability to specialize in a particular color or to a few colors. When a truck of one of the thirteen relatively infrequent colors arrives in Morley's system, there is a high probability that no queue has a truck of this color at the end. The result will be that this truck is assigned to the shortest queue with space. R-Wasps, on the other hand, allows one or more of the booths to

Table 2. Problem 2: Setup time is very significant in this problem.

	R-Wasps	Morley	Campos1	Campos2
Num. setups	265.33 ± 6.08	406.13 ± 9.54	480.12 ± 1.37	478.20 ± 1.66
Cycle time	26.72 ± 1.08	42.37 ± 6.48	136.47 ± 2.61	131.84 ± 3.31
Throughput	972.22 ± 2.11	873.69 ± 20.07	710.78 ± 3.79	713.84 ± 4.54
Queue length	2.97 ± 0.30	Overflow	Overflow	Overflow

95% confidence intervals are shown.

specialize in the color of high demand. So for example, let's consider that two or three of the booths have specialized in this single color of high demand and are unwilling to accept any of the other colors. Now when a truck requiring one of the less frequent colors arrives, even if the shortest queue is one of these two or three specialized booths, the system will assign it elsewhere. After all, if a setup is going to be accrued anyway and if there is such a high demand for the color these booths have specialized to with a high probability another truck of this highly demanded color will arrive soon, then it makes complete sense to route this less demanded color away from the specialized booths. Otherwise you are sure to accrue two setups rather than just a single setup. The fundamental difference between R-Wasps and either Morley's or Campos et al.'s is that Morley's paint booth agents (as well as Campos et al.'s agents) are required to bid as long as there is queue space; whereas any of our routing wasps can choose to ignore a stimulus and remain specialized even if there is space in its queue.

It can be further noted, by examining the average throughput and average queue lengths of the systems, that only R-Wasps is capable of handling the increased system load presented by the very large setup times. Morley, Campos1, and Campos2, all result in the queues as well as the "storage" location overflowing which has previously been considered by Morley and Campos et al. as a system failure. In this problem, in the worst case of every job requiring setup, the system can process 7 trucks in 13 minutes (i.e., 7 paint booths; 3 minutes process time plus 10 minutes setup per truck). During that 13 minutes, 13 new trucks would arrive (1 per minute). If this worst case behavior was to occur, then the system would quickly overflow the queues. Though Morley's system does not behave quite this poorly, 46.5% of the trucks require setup on average leading to each truck requiring an average of 7.65 minutes for painting and paint preparations. This means Morley's system processes 7 trucks every 7.65 minutes on average which is at a slower rate than the incoming stream of trucks resulting in an eventual overload of the system. Campos1 and Campos2 are even more extreme (e.g., Campos1 requires an average of 9.75 minutes for painting and paint preparation per truck). This is all compared to R-Wasps 5.73 minutes on average for painting and paint color change time per truck. One important thing to note here is that although R-Wasps does not overflow the queues for Problem 2, it comes very near doing so on average (a few runs actually do). This can be seen by looking at the average queue length of 2.97 which is very near the queue size limit of 3 trucks. So although R-Wasps uses on average only 5.73 minutes for painting and setup per truck, it achieves this by allowing some queues to grow to capacity (see the cycle time numbers) while leaving other machines with very short queues through specialization.

4.4. Unexpected machine breakdowns

To demonstrate robustness in the face of uncertain events, we now consider problems where paint booths may unexpectedly break down for some period of time. These problems also serve to examine if the adaptive systems – R-Wasps, Campos1, and Campos2 – are able to respond quickly enough to such events or if they suffer

from an over-adaptation to the system configuration prior to the unexpected events. We consider two such scenarios:

– Problem 3: Setup time is 1 minute as in Morley's original problem (Problem 1). Every simulated minute there is a probability of 0.05 that one randomly selected paint booth will breakdown for a period of time randomly chosen from the interval [1, 20]. When a booth is down, it does not accept additional trucks into its queue. A down booth also neither paints nor flushes paint from the system. Furthermore, any trucks already assigned to its queue must remain there until the booth is online again.
– Problem 4: Same as Problem 3 with the exception that the probability of some booth breaking down is increased to 0.1.

In Tables 3 and 4 we see the results of Problems 3 and 4, respectively. As you would expect, the average cycle times are higher for Problem 3 (probability of some booth breaking down is 0.05 during each simulated minute) as compared to Problem 1 (no paint booth breakdowns). Also as you would expect as you further increase the rate at which paint booths may break down in Problem 4, average cycle times continue to increase. This is all due to a decreased throughput capacity given that machines may occasionally go offline. The trends, however, remain the same within the comparison of the four systems that are under analysis. Also, as in Problems 1 and 2, R-Wasps is far superior in terms of the number of setups (paint flushes) as compared to the other three systems in these problems that include the possibility of unexpected paint booth breakdowns. The closest system in terms of this objective of minimizing the number of setups is Campos2. R-Wasps results in 8.75% fewer trucks requiring a

Table 3. Problem 3: During each simulated minute, there is a 0.05 probability that a randomly selected paint booth will break down for a period of time selected uniformly at random from the interval [1, 20].

	R-Wasps	Morley	Campos1	Campos2
Num. setups	291.80 ± 2.52	447.52 ± 3.57	520.8 ± 4.47	379.34 ± 3.72
Cycle time	9.26 ± 0.14	4.33 ± 0.05	3.80 ± 0.01	7.18 ± 0.04
Throughput	991.74 ± 0.42	996.73 ± 0.24	997.19 ± 0.15	993.82 ± 0.27
Queue length	0.23 ± 0.05	0.01 ± 0.01	0.00 ± 0.00	0.03 ± 0.02

95% confidence intervals are shown.

Table 4. Problem 4: During each simulated minute, there is a 0.1 probability that a randomly selected paint booth will break down for a period of time selected uniformly at random from the interval [1, 20].

	R-Wasps	Morley	Campos1	Campos2
Num. setups	296.34 ± 2.80	460.15 ± 3.63	528.55 ± 4.75	389.12 ± 3.57
Cycle time	11.24 ± 0.20	4.85 ± 0.06	4.15 ± 0.02	7.81 ± 0.05
Throughput	989.82 ± 0.68	996.20 ± 0.31	996.85 ± 0.18	993.62 ± 0.37
Queue length	0.48 ± 0.09	0.01 ± 0.01	0.00 ± 0.00	0.06 ± 0.03

95% confidence intervals are shown.

paint flush as compared to Campos2 in Problem 3; and it further results in 9.22% fewer trucks requiring a paint flush as compared to Campos2 in Problem 4. If you recall the results of Problem 1 where machines did not break down, R-Wasps resulted in 8.21% fewer trucks requiring a paint flush as compared to Campos2. The difference in performance between R-Wasps and Campos et al.'s system grows as the probability of machine breakdowns increases, suggesting that R-Wasps adapts more effectively to the unexpected machine breakdowns.

4.5. *Changing the color distribution*

We now demonstrate robustness to a change in the job mix with the following two scenarios:

- Problem 5: Same as in Morley's original problem (Problem 1), but with a change in the color distribution of arriving trucks. With probability 0.25, a truck requires color C1; and with probability 0.25, a truck requires color C2. Furthermore, a truck requires color C3 with probability $\frac{1}{24}$ (and likewise for each of colors C4, ..., C14).
- Problem 6: This problem has a dynamically changing color distribution. For the first 300 simulation minutes, the distribution is as in Problem 1. For the following 400 simulation minutes, the distribution is as in Problem 5. For the last 300 simulation minutes, the distribution is as in Problem 1 but with the arrival rates of colors C1 and C2 reversed (i.e., $P(C2) = 0.5$ and $P(C1) = \frac{1}{26}$). All other problem characteristics are as in Problem 1.

Tables 5 and 6 show the results of Problems 5 and 6, respectively. It should be noted that the parameters of the four algorithms compared are left the same (i.e., tuned to Morley's original problem). They could have been re-tuned to the color distributions of Problems 5 and 6, but part of the purpose of these experiments is to demonstrate robustness to the color distribution, including in the face of dynamic changes to the job mix. In such instances, re-tuning the system parameters may be too costly.

For Problem 5 (Table 5), each of the algorithms perform very similarly in terms of cycle time to how they performed on Problem 1. In terms of the primary objective of minimizing the number of setups (i.e., paint system flushes), Morley and Campos1 both give results that are worse than in Problem 1. An interesting thing to note is

Table 5. Problem 5: Same as in Problem 1, but with a change to the color distribution.

	R-Wasps	Morley	Campos1	Campos2
Num. setups	277.99 ± 2.40	516.83 ± 3.52	570.43 ± 3.28	334.31 ± 5.06
Cycle time	8.00 ± 0.10	3.82 ± 0.02	3.57 ± 0.003	5.55 ± 0.06
Throughput	993.17 ± 0.30	997.18 ± 0.17	997.43 ± 0.10	995.45 ± 0.24
Queue length	0.07 ± 0.03	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00

95% confidence intervals are shown.

Table 6. Problem 6: Same as in Problem 1, but with a dynamically changing color distribution.

	R-Wasps	Morley	Campos1	Campos2
Num. setups	284.54 ± 2.62	470.25 ± 3.31	532.98 ± 4.25	351.90 ± 3.78
Cycle time	8.00 ± 0.10	3.85 ± 0.03	3.53 ± 0.004	6.09 ± 0.04
Throughput	993.13 ± 0.38	997.00 ± 0.21	997.55 ± 0.10	994.43 ± 0.20
Queue length	0.10 ± 0.03	0.00 ± 0.00	0.00 ± 0.00	0.01 ± 0.01

For the first 300 simulation minutes, the distribution is as in Problem 1. For the following 400 simulation minutes, it is as in Problem 5. For the last 300 minutes, the distribution is as in Problem 1, but with the frequencies of colors C1 and C2 switched. 95% confidence intervals are shown.

that R-Wasps and Campos2 actually improve their performance on the objective of minimizing number of setups as compared to their respective performances on Problem 1. This is especially interesting since: (1) their respective system parameters were tuned for Problem 1; and (2) the color distribution in Problem 5 is somewhat more diverse than it had been for Problem 1 which would intuitively make the objective of minimizing the number of setups more difficult to achieve. Overall, R-Wasps is again superior to the other three algorithms on this objective of minimizing number of setups.

For Problem 6 (Table 6), the performance of each of the algorithms in terms of the objective of minimizing the number of setups lies someplace between the algorithms' respective performances on Problems 1 and 5. This seems to intuitively make sense since part of Problem 6 is as in Problem 1, part of Problem 6 is as in Problem 5, and part of Problem 6 is as in Problem 1 but with a different high demand color. So in a sense, the dynamically changing job mix of Problem 6 is a cross between Problems 1 and 5. This is of course not exactly true since whenever the color mix changes there may be trucks in queues, booths may be in the process of painting trucks, etc., but it is sufficiently true to aid in understanding the results. In any event, the trends are again the same as in the other problems. R-Wasps clearly dominates the others in terms of the objective of minimizing the number of paint system flushes (number of setups) and is clearly robust to dynamic changes in job mix (color distribution).

5. System analysis

In this section we examine the behavior of the routing wasps on a few simple factory configurations. The purpose of this analysis is to illustrate that the behavior adapted by the routing wasps corresponds to what intuitively is the "best" routing policy. In Section 5.1 we describe the design of the problems used in this analysis. The behavior of the routing wasps is analyzed in Section 5.2 using plots of the average response thresholds over time.

5.1. Experimental design

The analysis of system behavior in this section considers factories which produce two products (henceforth, Job Type A and Job Type B) and multi-purpose machines that

can process either of the two product types. We consider a factory configuration with two such machines as well as one with four such machines. This is a scaled down version of the real-world problem of Section 4 (which had 14 job types, or colors, and 7 machines) to allow for a more readily to comprehend analysis.

Setup time to reconfigure a machine for the alternate job type is 30 time units. Process time of a job is equal to 15 plus a Gaussian noise factor with mean 0.0 and standard deviation 1.0. The resulting process time is taken to the next higher integer value for times greater than 15 and taken to the next lower integer value for times less than 15. The overall process time is bounded in the interval of 10–20, inclusive. There is no limit in the length of the job queues (unlike the real-world example of Section 4).

Jobs are released to the factory floor dynamically according to four different product mixes (3 static and 1 changing). In each, arrival rates are defined by the probability that a new job of each type is released during a given time unit. The arrival rates for the two machine problems are as follows:

- 50/50 mix: $P(\text{Job Type A}) = 0.05$, $P(\text{Job Type B}) = 0.05$
- 85/15 mix: $P(\text{Job Type A}) = 0.0857$, $P(\text{Job Type B}) = 0.0143$
- 100/0 mix: $P(\text{Job Type A}) = 0.133$, $P(\text{Job Type B}) = 0.0$
- Changing mix: For the first half of the simulation $P(\text{Job Type A}) = 0.0857$, $P(\text{Job Type B}) = 0.0143$, then for the second half of the simulation $P(\text{Job Type A}) = 0.0143$, $P(\text{Job Type B}) = 0.0857$.

To get the rates for the four machine problems simply multiply these rates by 2. These arrival rates correspond approximately to medium-to-heavily loaded factories.

The values of the various parameters of the system are the following: $\theta_{\min} = 1$, $\theta_{\max} = 1000$, $\delta_1 = 2$, $\delta_2 = 1$, and $\delta_3 = 1.001$. The analysis considers an average of 100 runs with different arrival sequences according to the distributions above. The simulations are 5000 time units in length.

5.2. *Response threshold analysis*

In Figures 3 and 4, we see plots of the response thresholds of the routing wasps for various job mixes and two machines. The 50/50 and 100/0 job mixes are in Figure 3 and the 85/15 and changing job mixes are in Figure 4. The plots indicate the adaptation of the response thresholds over time. There are two plots for each job mix – one for each job type. Each plot contains the response thresholds to the given type for each of the machines.

In the 50/50 job mix, we see that each machine specializes to a different job type. In other words, one machine adapts a preference to job type A (very low response threshold to type A, virtually a flat line in the plots at $\theta_{\min}$, and a high response threshold to type B); while the other machine similarly adapts a preference for type B jobs.

In the 100/0 job mix, we see that both machines quickly adapt their configurations to that associated with the single job type in the system. In other words, both

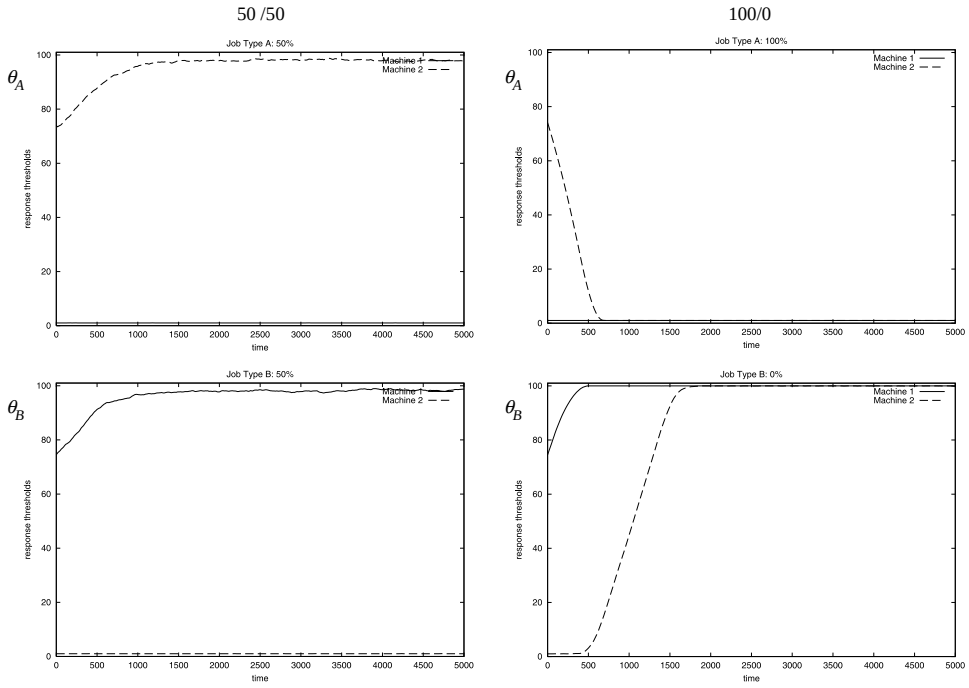

Figure 3. Plots of the average response thresholds over time of the routing wasps for the 50/50 and 100/0 job mixes and two machines.

machines adapt their response thresholds as to specialize in type A jobs (very low response thresholds); while at the same time adjusting the response thresholds for type B jobs to very large values, signifying a disinterest in those jobs.

For the 85/15 job mix, both machines are willing to take jobs of the type of high demand (job type A); while only one is interested in the type B jobs. That is, both machines adapt low response thresholds to type A jobs; while only one of the machines adapts a relatively low response threshold to type B jobs.

The first half of the changing job mix simulations corresponds to that of the 85/15 job mix; while in the second half we see the machines changing roles to handle the new 15/85 mix. The first half of the response threshold plots for this changing job mix is virtually identical to those of the 85/15 job mix. And then the response thresholds begin adjusting into a configuration that signifies that only one machine is bidding on the type A jobs when the job mix changes; while both machines begin bidding on jobs of type B.

If we examine similarly the four machine problems in Figures 5 and 6 we find the same sort of behavior. That is, in the 100/0 job mix all machines specialize in the single job type, in the 50/50 job mix half of the machines specialize to each job type, and in the 85/15 job mix all machines are willing to take the job type of higher demand while only one has a strong interest in the other job type.

This corresponds, intuitively, to the behavior the system should exhibit for "optimal" performance. That is, attempt to specialize a proportion of machines

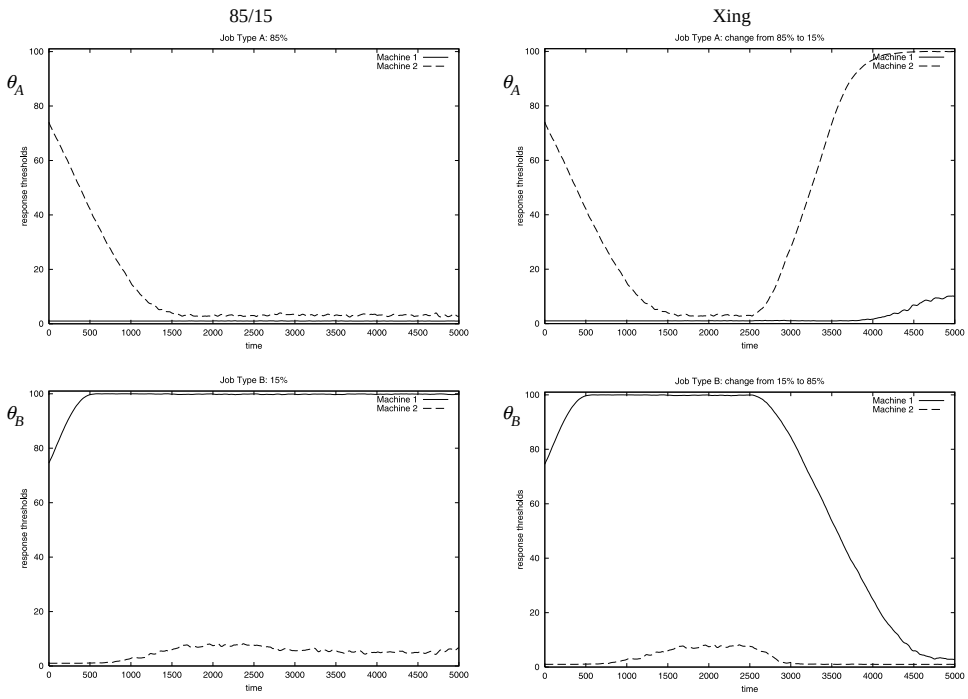

Figure 4. Plots of the average response thresholds over time of the routing wasps for the 85/15 and dynamically changing job mixes and two machines.

according to the proportion of jobs of each type (where possible). In the 100/0 mix, 100% of the jobs are of type A so 100% of the machines should specialize to type A jobs. In the 50/50 mix, we should similarly see 50% of the machines specialized to type A and 50% specialized to type B. The 85/15 mix illustrates a somewhat trickier issue. That is, it is not possible to exactly specialize the appropriate proportion of machines to each type. In the two machine case, only one machine specializes in the type B jobs; while both specialize to the type A jobs. One of these latter, however, specializes slightly less strongly in the type A jobs (i.e., the machine that is also bidding on the type B jobs). And we see similar behavior in the four machine problem for the machine specialized in the type B jobs.

One thing that is worth noting, particularly in regard to the 85/15 job mix and the changing job mix, is that the system requires some time to converge to a stable behavior. For example, if you examine the 85/15 job mix response threshold plots in Figure 4, you can see that both machines are taking jobs of the product of highest demand only after (approximately) time unit 1000. This is similarly true in the changing job mix (also in Figure 4). With the changing job mix, however, we have the addition of a second stage of adaptation beginning at time unit 2500 when the job mix changes drastically from 85/15 to 15/85. Part of the reason this adaptation rate appears so slow is due to the infinite maximum queue length constraint used in this analysis. While the machine specialized to only the formerly highly demanded

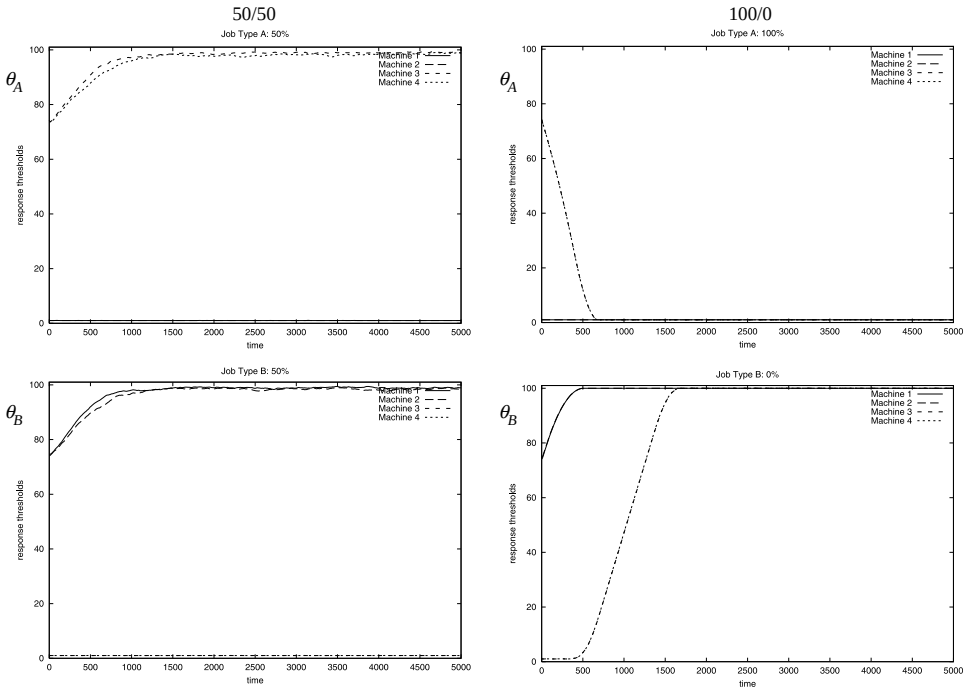

Figure 5. Plots of the average response thresholds over time of the routing wasps for the 50/50 and 100/0 job mixes and four machines.

job type (type A) is readjusting its response thresholds to allow for bidding on type B jobs (the new type of higher demand), the other machine which was already specialized to type B is the only machine bidding on the type B jobs and thus is awarded these jobs given the lack of a finite queue length restriction. Figure 7 shows plots of the response thresholds for the two machine changing mix problem, but this time with a queue size limit of three jobs. If you compare this to the changing mix problem of Figure 4, you will see that the finite queues result in a noticeable improvement in the adaptation rate of response thresholds.

6. Limitations

6.1. Rate of adaptation

Our routing wasps require some amount of time to adapt to the product mix as well as to re-adapt to changing product demands. In the analysis of behavior in Section 5, this adaptation time appeared to be somewhat serious with respect to the changing job mix problem. It is not quite as serious as it at first appears. If you recall, the routing wasps did not appear to suffer during adaptation time in the real-world problem of Section 4 in that there was only a slight downgrade in performance in the

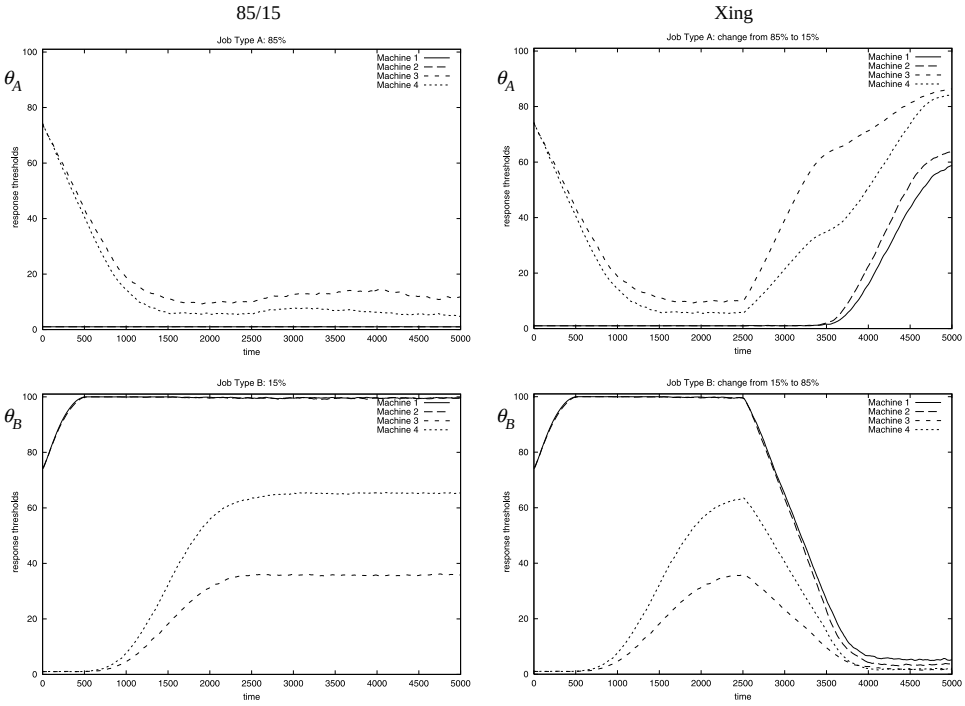

Figure 6. Plots of the average response thresholds over time of the routing wasps for the 85/15 and dynamically changing job mixes and four machines.

changing product mix problems as compared to the static product mix problems. The reason we did not observe poor performance during adaptation time in this problem was the finite queue limit constraint. In Section 5, there was no queue limit and during the initial adaptation to the 85/15 mix and the re-adaptation in the changing mix one of the machines ends up over-specializing and building up a large queue size until the other machine adapts its response thresholds appropriately. The queue limit constraint prevents such drastic over-specialization. This constraint is also very realistic and characteristic of real-world problems. For example, in many real-world problems, the length of the queue is constrained due to factory characteristics. In the vehicle paintshop problem. for example, there is only so much space to maintain a line of trucks due to their physical size and the physical size of the paintshop. However, although this constraint is realistic, it may still be desirable to overcome the rate of adaptation limitation in the event of dealing with a system with no such queue length limit constraint (or a very large limit). A couple of possibilities for dealing with this limitation include: (1) maintaining some statistics on the job mix and boosting the values of the learning parameters δ_i if a large enough job mix change has been detected; (2) incorporating an additional response threshold update rule that is triggered when a machine is idle and has an empty queue and its routing wasp receives a stimulus from a job but does not bid; and (3) incorporating queue length into the stochastic rule used by the routing wasps in determining whether or

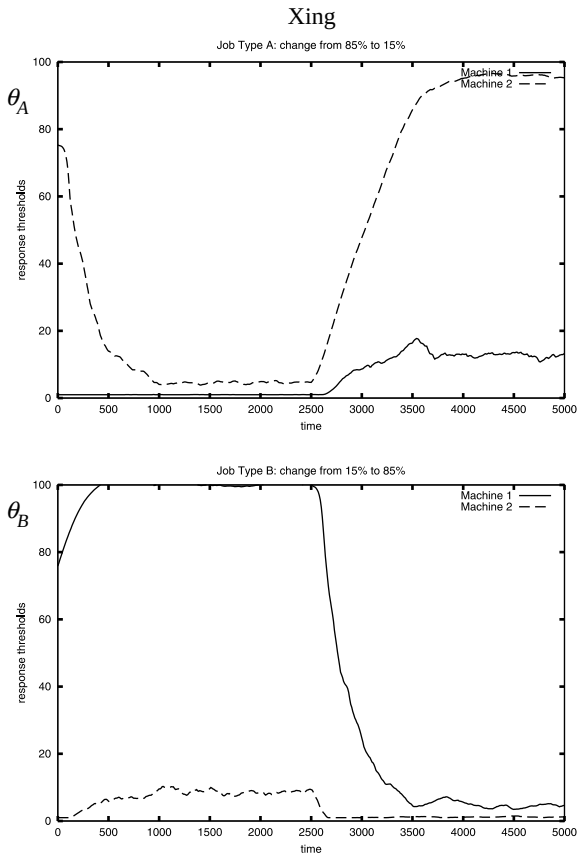

Figure 7. Plots of the average response thresholds over time of the routing wasps for the dynamically changing job mix, finite queue length, and two machines.

not to bid on a job. The first of these potential solutions is domain specific; whereas the second and third solutions appear to be more generally applicable to other problems as well as a more cohesive fit with the underlying model. There are perhaps other equally desirable options.

6.2. *Parameter tuning*

The response threshold formulation involves a number of parameters. The experiments presented in this paper used parameter values that were tuned by hand. This process can be enhanced with a more sophisticated automated meta-level optimization of the parameters such as was used by Morley to optimize the parameters of his system. There has been a great deal of work on the meta-level optimization of the control parameters of genetic algorithms (e.g., [12, 14, 27, 58]) and perhaps a similar approach would benefit our routing wasps.

7. Related work

There has been much research in recent years into distributed and agent-based approaches to scheduling problems. One agent coordination paradigm that appears to be quite popular among the researchers in this area is that of artificial market systems. In the various market-based and auction approaches that appear in the literature, agents representing resources use bidding mechanisms of one sort or another to coordinate the assignment of required resources to tasks or jobs (e.g., [21, 28, 33, 37, 38, 41, 42, 47, 56, 57]). There is evidence that market-based approaches can produce globally optimal or near-optimal solutions. For example, Walsh et al. [56] show that for discrete resource allocation problems an equilibrium solution is an optimal solution. However, this characteristic of auction-based approaches is not a cure-all. The problem is that equilibrium solutions do not always exist; and often when they do exist, the problem of finding the equilibrium is NP-hard. In any case, bidding mechanisms appear to be fairly closely related to the stimulus-response mechanism of the wasp behavior model as pointed out in [4]. That is, high bids correspond to low response thresholds; and low bids correspond to high response thresholds. However, the relation of bidding mechanisms to the wasp model goes beyond the simple analogy made by Bonabeau et al. If coupled with a market-based system, the stimulus-response mechanism of wasps can impart on the agents the ability to simply not bid and to ignore a "call for bids" if the stimulus is sufficiently below the response threshold even if it is capable of performing the requested task as is done by our routing wasps. More importantly, it allows for a mechanism for the adaptation of a decision policy regarding when to bid and when to not bid.

Nouyan has recently made a few promising extensions to our system [44]. Nouyan's work is an extension of an earlier version of our system [18]. The problems Nouyan considers are somewhat simpler than those considered in this paper and do not consider unexpected events such as machine breakdowns. Two promising modifications to our system suggested by Nouyan include additional response threshold update rules: (1) for each unallocated job, decreasing the response thresholds for its type for all machine agents; (2) decreasing the response threshold of an idle machine for the type of a job for which it did not bid. Nouyan's additional update rules focus on improving the rate of adaptation of response thresholds in the model.

Another distributed scheduling approach that has been gaining in popularity in the evolutionary computation community in recent years builds from the Ant Colony Optimization (ACO) metaheuristic of Dorigo et al. [24–26]. ACO uses a population of ant-like agents that communicate indirectly via trail laying and following to build solutions to the scheduling problem at hand. There have been numerous applications of ACO to various scheduling problems including: the sequential ordering problem [31], job shop scheduling [55], flow shop scheduling [50], vehicle routing [9, 32], bus driver scheduling [30], tardiness scheduling problems [1, 22], and resource-constrained project scheduling [40]. However, although all of these applications of ACO to scheduling problems use a population of agents to solve their respective problems, all of these systems build their solutions in advance. So although their solutions are computed in a distributed manner, they deal with a "static" problem rather than the type of dynamic problem that we have considered

in this paper. Based on the ACO paradigm, our previous attempt at a distributed solution to this dynamic problem was that of AC^2 [15]. However, it suffered from frequent convergence to sub-optimal equilibrium in a game-theoretic sense [13]. A more successful application of ACO to a dynamically changing problem has been to the problem of network routing. Schoonderwoerd et al. [48, 49] have developed an effective system called Ant Based Control (ABC) for adapting routing tables in circuit-switched networks based on the ACO framework. Similarly, Di Caro and Dorigo [23] have developed a system called AntNet in which artificial ants adapt the routing tables of packet-switched networks.

Another aspect of our research not discussed in this paper but closely related is a framework for randomization of dispatch scheduling heuristics based on the model of wasp social hierarchy formation (see [17, 19, 20]). This framework uses a population of what we call scheduling wasps that interact with each other to prioritize jobs waiting in a queue. The result is an effective stochastic mechanism for amplifying the performance of dispatch scheduling policies in cases when the deterministic policy is less informed. For hard instances of the dynamic scheduling objective of minimizing weighted tardiness under sequence-dependent setup constraints, our stochastic framework of the scheduling wasps has been shown superior to state-of-the-art deterministic dispatch policies for the problem [17, 20].

8. Conclusion

In this paper we have presented an adaptive multi-agent system for dynamic factory routing based on various aspects of a computational model of the adaptive behavior observed in wasp colonies. In our system, the assignment of jobs to machines is performed by computational agents called routing wasps in a manner analogous to task allocation among real wasps. The routing wasps decide whether or not to bid for an arriving job stochastically according to its response threshold to the type of job. Specifically we dealt with the problem of assigning trucks to paint booths in a simulation of a dynamic vehicle paintshop. In the event more than one routing wasp attempts to assign the same job to their respective machines (i.e., bids for the new truck), our system employs a model of self-organized social hierarchies within wasp colonies to decide among the competing routing wasps via a tournament of dominance contests.

Experimentally, in a simulated factory environment, we compared our adaptive wasp-like agents to the "real world-proven" bidding mechanism of Morley that was originally designed for this vehicle paintshop problem. We also compared our system to the related system of Campos et al. which utilized the same model of wasp behavior to define an adaptive bid determination rule. R-Wasps (our system) was demonstrated to be superior to the others in terms of the overall objective of minimizing the number of paint flushes required by the system. The robustness of our system in the face of unexpected machine breakdowns, as well as to changes in the color distribution, was further demonstrated and our system continued to stand out as superior to the benchmark systems. Through the adaptive ability of machines to specialize in one or a few product types (e.g., in this specific case truck colors)

R-Wasps is able to robustly minimize costs associated with sequence-dependent constraints (e.g., minimize the cost associated with paint wasted during system paint flushes).

More generally, we believe that our multi-agent model offers a flexible, decomposable approach to coordinating material flows to meet changing demands and other dynamic constraints. As such, it should also be naturally applicable to more global supply-chain coordination problems. With continuing trends toward specialization on core competencies, manufacturing organizations must rely increasingly on coupling their respective capabilities and partnering to capitalize on new market opportunities, and the ability to rapidly and dynamically reconfigure supply chains becomes increasingly important. This is one direction of our continuing research.

Acknowledgments

This work has been funded in part by the Department of Defense Advanced Research Projects Agency and the US Air Force Rome Research Laboratory under contracts F30602-97-2-0066 and F30602-00-2-0503 and by the CMU Robotics Institute. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force or US Government.

Notes

1. For a survey focused on manufacturing applications of social insect behavior see Cicirello and Smith [16].
2. In this definition of force, the "stronger" wasp is the wasp with the smaller force. This may seem counter-intuitive with the usual connotation of the word "force", but defining force in this way is cleaner mathematically. Perhaps "weakness" may have been a more accurate term to use rather than "force", but we chose the latter to correspond more closely to the terminology of the model of real wasp behavior.
3. We began with an arbitrary parameter set and performed a gradient search. That is, we examined the performance of the initial set. Next, we tweaked the value of the first parameter until no improvement could be made. Then we continued in this manner for each of the system parameters.
4. Cycle time is the length of time from the arrival of a job to its completion time.
5. Throughput is the number of jobs completed by the time the simulation ends.
6. The average queue length per machine is taken when the simulation ends and refers to the queues only. Jobs in process or in setup are not included in this computation.

References

1. A. Bauer, B. Bullnheimer, R. F. Hartl, and C. Strauss, "An ant colony optimization approach for the single machine total tardiness problem," in *CEC99: Proceedings of the Congress on Evolutionary Computation*, 1999, pp. 1445–1450.
2. T. Beaumariage and K. Kempf, "Attractors in manufacturing systems with chaotic tendencies," Presentation at INFORMS-95, New Orleans, http://www.informs.org/Conf/NewOrleans95/TALKS/TB07.3.html, 1995.

3. R. Beckers, O. E. Holland, and J. L. Deneubourg, "From local actions to global tasks: Stigmergy and collective robotics," in R. A. Brooks and P. Maes, (eds.), *Artificial Life IV: Proceedings of the Fourth International Workshop on the Synthesis and Simulation of Living Systems*, 1994, pp. 181–189.
4. E. Bonabeau, M. Dorigo, and G. Theraulaz, *Swarm Intelligence: From Natural to Artificial Systems*, Santa Fe Institute Studies in the Sciences of Complexity, Oxford University Press: Oxford, 1999.
5. E. Bonabeau, M. Dorigo, and G. Theraulaz, "Inspiration for optimization from social insect behaviour," *Nature*, vol. 406, pp. 39–42, 2000.
6. E. Bonabeau, A. Sobkowski, G. Theraulaz, and J. L. Deneubourg, "Adaptive task allocation inspired by a model of division of labor in social insects," in D. Lundh and B. Olsson, (eds.): *Bio Computation and Emergent Computing*, World Scientific, pp. 36–45, 1997.
7. E. Bonabeau, G. Theraulaz, and J. L. Deneubourg, "Fixed response thresholds and the regulation of division of labor in insect societies," *Bull. Math. Biol.*, vol. 60, pp. 753–807, 1998.
8. J. Braslaw, Personal communication, Material Sciences Department, Ford Research, 2001.
9. B. Bullnheimer, R. F. Hartl, and C. Strauss, "An improved ant system algorithm for the vehicle routing problem," *Ann. Oper. Res.*, vol. 89, pp. 319–328, 1999.
10. S. Bussmann, "Agent-oriented programming of manufacturing control tasks," in *Proceedings of the 3rd International Conference on Multi-Agent Systems (ICMAS-1998)*, 1998, pp. 57–63.
11. M. Campos, E. Bonabeau, G. Théraulaz, and J. Deneubourg, "Dynamic scheduling and division of labor in social insects," *Adapt. Behav.*, vol. 8, no. 2, pp. 83–96, 2000.
12. Y. J. Cao and Q. H. Wu, "Optimization of control parameters in genetic algorithms: A stochastic approach," *Int. J. Syst. Sci.*, vol. 30, no. 5, pp. 551–559, 1999.
13. V. A. Cicirello, "A game-theoretic analysis of multi-agent systems for shop floor routing," Technical Report CMU-RI-TR-01-28, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, 2001.
14. V. A. Cicirello and S. F. Smith, "Modeling GA performance for control parameter optimization," in D. Whitley, D. Goldberg, E. Cant-Paz, L. Spector, I. Parmee, and H. Beyer, (eds.), *GECCO-2000: Proceedings of the Genetic and Evolutionary Computation Conference*, Las Vegas, NV, 2000, pp. 235–242.
15. V. A. Cicirello and S. F. Smith, "Ant colony control for autonomous decentralized shop floor routing," in *ISADS-2001: International Symposium Autonomous Decentralized Systems*, Dallas, TX, 2001, pp. 383–390.
16. V. A. Cicirello and S. F. Smith, "Insect societies and manufacturing," in *The IJCAI-01 Workshop on Artificial Intelligence and Manufacturing, Working Notes*, Seattle, WA, 2001, pp. 33–38.
17. V. A. Cicirello and S. F. Smith, "Randomizing dispatch scheduling policies," in *Using Uncertainty Within Computation: Papers from the 2001 AAAI Fall Symposium, Technical Report FS-01-04*, North Falmouth, Massachusetts, 2001, pp. 30–37.
18. V. A. Cicirello and S. F. Smith, "Wasp-like agents for distributed factory coordination," Technical Report CMU-RI-TR-01-39, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, 2001.
19. V. A. Cicirello and S. F. Smith, "Wasp nests for self-configurable factories," in J. P. Müller, E. Andre, S. Sen, and C. Frasson, (eds.), *Proceedings of the Fifth International Conference on Autonomous Agents*, Montreal, Quebec, Canada, 2001, pp. 473–480.
20. V. A. Cicirello and S. F. Smith, "Amplification of search performance through randomization of heuristics," in *The Eighth International Conference on Principles and Practice of Constraint Programming (CP-02)*, Ithaca, NY, 2002.
21. J. Collins, M. Tsvetovatyy, B. Mobasher, and M. Gini, "MAGNET: A multi-agent contracting system for plan execution," in G. F. Luger, (ed.), *Proceedings of the Artificial Intelligence and Manufacturing Research Planning Workshop*, pp. 63–68, 1998.
22. M. den Besten, T. Stützle, and M. Dorigo, "Ant colony optimization for the total weighted tardiness problem," in M. Schoenauer, K. Deb, G. Rudolph, X. Yao, E. Lutton, J. J. Merelo, and H. S. Schwefel, (eds.), *Proceedings of PPSN-VI, Sixth International Conference on Parallel Problem Solving from Nature, Lecture Notes in Computer Science*, vol. 1917, 2000, pp. 611–620.
23. G. Di Caro and M. Dorigo, "AntNet: Distributed stigmergetic control for communications networks," *J. Artif. Intell. Res.*, vol. 9, pp. 317–365, 1998.
24. M. Dorigo and G. Di Caro, "The ant colony optimization meta-heuristic," in D. Corne, M. Dorigo, and F. Glover, (eds.), *New Ideas in Optimization*, McGraw-Hill: New York, 1999, pp. 11–32.

25. M. Dorigo and L. M. Gambardella, "Ant colony system: A cooperative learning approach to the traveling salesman problem," *IEEE Trans. Evolut. Comput.*, vol. 1, no. 1, pp. 53–66, 1997.
26. M. Dorigo, V. Maniezzo, and A. Colorni, "Ant system: Optimization by a colony of cooperating agents," *IEEE Trans. Syst. Man Cyber. – Part B: Cyber.*, vol. 26, no. 1, pp. 29–41, 1996.
27. A. E. Eiben, R. Hinterding, and Z. Michalewicz, "Parameter control in evolutionary algorithms," *IEEE Trans. Evolut. Comput.*, vol. 3, no. 2, pp. 124–141, 1999.
28. K. Fischer, J. P. Müller, M. Pischel, and D. Schier, "A model for cooperative transportation scheduling," in *ICMAS-95: Proceedings of the First International Conference on Multi-Agent Systems*, 1995, pp. 109–116.
29. T. D. Fitzgerald and S. C. Peterson, "Cooperative foraging and communication in caterpillars," *BioScience*, vol. 38, no. 1, pp. 20–25, 1988.
30. P. Forsyth and A. Wren, "An ant system for bus driver scheduling," Technical Report 97.25, University of Leeds, School of Computer Studies, Presented at the 7th International Workshop on Computer-Aided Scheduling of Public Transport, Boston, July 1997.
31. L. M. Gambardella and M. Dorigo, "Ant colony system hybridized with a new local search for the sequential ordering problem," *INFORMS J. Comput.*, vol. 12, no. 3, pp. 237–255, 2000.
32. L. M. Gambardella, E. Taillard, and G. Agazzi, "MACS-VRPTW: A multiple ant colony system for vehicle routing problems with time windows," in D. Corne, M. Dorigo, and F. Glover, (eds.), *New Ideas in Optimization*, McGraw-Hill: New York, 1999, pp. 63–76.
33. S. Y. Goldsmith and L. D. Interrante, "An autonomous manufacturing collective for job shop scheduling," in G. F. Luger, (ed.), *Proceedings of the Artificial Intelligence and Manufacturing Research Planning Workshop*, 1998, pp. 69–74.
34. D. M. Gordon, "The organization of work in social insect colonies," *Nature*, vol. 380, pp. 121–124, 1996.
35. K. Kempf and T. Beaumariage, "Chaotic behavior in manufacturing systems," in *AAAI-94 Workshop Program, Reasoning About the Shop Floor, Workshop Notes*, 1994, pp. 82–96.
36. W. H. Kirchner and W. F. Towne, "The sensory basis of the honeybee's dance language," *Sci. Am.*, vol. 270, no. 6, pp. 74–80, 1994.
37. K. Kuwabara and T. Ishida, "Equilibratory approach to distributed resource allocation: toward coordinated balancing," in C. Castelfranchi and E. Werner, (eds.), *Artificial Social Systems: 4th European Workshop on Modelling Autonomous Agents in a Multi-Agent World (Selected Papers)*, 1992, pp. 133–146.
38. G. Y. J. Lin and J. J. Solberg, "Integrated shop floor control using autonomous agents," *IIE Trans.*, vol. 24, no. 3, pp. 57–71, 1992.
39. J. S. Liu, "Coordination of multiple agents in distributed manufacturing scheduling," Ph.D. thesis, The Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, 1996.
40. D. Merkle, M. Middendorf, and H. Schmeck, "Ant colony optimization for resource-constrained project scheduling," in *GECCO-2000: Proceedings of the Genetic and Evolutionary Computation Conference*, 2000, pp. 893–900.
41. D. Morley, "Painting trucks at general motors: The effectiveness of a complexity-based approach," in *Embracing Complexity: Exploring the Application of Complex Adaptive Systems to Business*, The Ernst and Young Center for Business Innovation, 1996, pp. 53–58.
42. D. Morley and C. Schelberg, "An analysis of a plant-specific dynamic scheduler," in *Proceedings of the NSF Workshop on Dynamic Scheduling*, 1993, pp. 115–122.
43. T. E. Morton and D. W. Pentico, *Heuristic Scheduling Systems: With Applications to Production Systems and Project Management*, John Wiley and Sons, 1993.
44. S. Nouyan, "Agent-based approach to dynamic task allocation," in M. Dorigo, G. Di Caro, and M. Sampels, (eds.), *Ant Algorithms: Third International Workshop, ANTS 2002, Proceedings, Lecture Notes in Computer Science*, vol. LNCS 2463, 2002, pp. 28–39.
45. P. S. Ow, S. F. Smith, and R. Howie, "A cooperative scheduling system," in M. D. Oliff, (ed.), *Expert Systems and Intelligent Manufacturing*, Elsevier Science Publishing Co., Inc., 1988, pp. 43–56.
46. V. Parunak, A. Baker, and S. Clark, "The AARIA agent architecture: From manufacturing requirements to agent-based system design," in *Proceedings of the ICAA'98 Workshop on Agent-Based Manufacturing*, 1998.

47. R. J. Rabelo and L. M. Camarinha-Matos, "Negotiation in multi-agent based dynamic scheduling," *Robot. Comput.-Integrat. Manuf.*, vol. 11, no. 4, pp. 303–309, 1994.
48. R. Schoonderwoerd, O. Holland, and J. Bruten, "Ant-like agents for load balancing in telecommunications networks," in *Agents '97, Proceedings of the First International Conference on Autonomous Agents*, 1997, pp. 209–216.
49. R. Schoonderwoerd, O. Holland, J. Bruten, and L. Rothkrantz, "Ant-based load balancing in telecommunications networks," *Adapt. Behav.*, vol. 5, no. 2, pp. 169–207, 1997.
50. T. Stützle, "An ant approach to the flow shop problem," in *Proceedings of the 6th European Congress on Intelligent Techniques & Soft Computing (EUFIT'98)*, vol. 3, pp. 1560–1564, 1998.
51. K. P. Sycara, S. F. Roth, N. Sadeh, and M. S. Fox, "Resource allocation in distributed factory scheduling," *IEEE Expert*, vol. 6, no. 1, pp. 29–40, 1991.
52. G. Theraulaz, E. Bonabeau, and J. L. Deneubourg, "Self-organization of hierarchies in animal societies: The case of the primitively eusocial wasp polistes dominulus christ," *J. Theor. Biol.*, vol. 174, pp. 313–323, 1995.
53. G. Theraulaz, E. Bonabeau, and J. L. Deneubourg, "Response threshold reinforcement and division of labour in insect societies," *Proc. R. Soc. London B*, vol. 265, no. 1393, pp. 327–335, 1998.
54. G. Theraulaz, S. Goss, J. Gervet, and J. L. Deneubourg, "Task differentiation in polistes wasp colonies: A model for self-organizing groups of robots," in *From Animals to Animats: Proceedings of the First International Conference on Simulation of Adaptive Behavior*, 1991, pp. 346–355.
55. S. van der Zwaan and C. Marques, "Ant colony optimisation for job shop scheduling," in *Proceedings of the '99 Workshop on Genetic Algorithms and Artficial Life GAAL'99*, 1999.
56. W. E. Walsh, M. P. Wellman, P. R. Wurman, and J. K. MacKie-Mason, "Some economics of market-based distributed scheduling," in *Proceedings of the Eighteenth International Conference on Distributed Computing Systems*, 1998, pp. 612–621.
57. M. Wellman, "A general-equilibrium approach to distributed transportation planning," in *AAAI-92: Proceedings of the Tenth National Conference on Artificial Intelligence*, 1992, pp. 282–289.
58. S. J. Wu and P. T. Chow, "Genetic algorithms for nonlinear mixed discrete-integer optimization problems via meta-genetic parameter optimization," *Eng. Optim.*, vol. 24, no. 2, pp. 137–159, 1995.