6 AntNet: An ACO Algorithm for Data Network Routing

An estimated lower bound on the size of the indexable Web is 320 million pages.
—Steve Lawrence and C. Lee Giles *Science*, *280*, 1998

Number of web pages indexed by Google in June 2003: more than 3 billion.
—www.google.com

In this chapter we discuss AntNet, an ACO algorithm designed to help solve the routing problem in telecommunications networks. Network routing refers to the activities necessary to guide information in its travel from source to destination nodes. It is an important and difficult problem. Important because it has a strong influence on the overall network performance. Difficult because networks' characteristics, such as traffic load and network topology, may vary stochastically and in a time-varying way. It is in particular these characteristics of the problem, in addition to the physical distributedness of the overall problem on a real network, that make ACO algorithms a particularly promising method for its solution. In fact, the ACO processing paradigm is a good match for the distributed and nonstationary (in topology and traffic patterns) nature of the problem, presents a high level of redundancy and fault-tolerance, and can handle multiple objectives and constraints in a flexible way.

Although AntNet is not the only ACO algorithm developed for routing problems, and not even the historically first one, we focus on it because it is the sole algorithm to have reached, at least at the experimental/simulation level at which it was tested, state-of-the-art performance. We give a detailed description of AntNet's data structures and control procedures, and a brief overview of the results obtained using a network simulation environment.

6.1 The Routing Problem

Communications networks can be classified as either circuit-switched or packet-switched. The typical example of a circuit-switched network is the telephone network, in which a virtual or physical circuit is set up at the communication start and remains the same for the communication duration. Differently, in packet-switched networks, also called *data networks*, each data packet can, in principle, follow a different route, and no fixed virtual circuits are established. In this case the typical examples are local area computer networks and the Internet.

Arguably, the main function of a data network, on which we focus in this chapter, is to assure the efficient distribution of information among its users. This can be achieved through the exploitation of an adequate network control system. One of the most important components of such a system, in conjunction with the admission, flow, and congestion control components, is routing (Walrand & Varaiya, 1996). Routing refers to the distributed activity of building and using *routing tables*. The routing table is a common component of all routing algorithms: it holds the information used by the algorithm to make the local forwarding decisions. The type of information it contains and the way this information is used and updated strongly depend on the algorithm's characteristics. One routing table is maintained by each node in the network: it tells the node's incoming data packets which among the outgoing links to use to continue their travel toward their destination node. One of the most distinctive aspects of the network routing problem is the nonstationarity of the problem's characteristics. In particular, the characteristics of traffic over a network change all the time, and in some important cases (e.g., the Internet) the traffic can fluctuate in ways difficult to predict. Additionally, the nodes and links of a network can suddenly go out of service, and new nodes and links can be added at any moment. Therefore, network routing is very different from the $\mathcal{NP}$-hard problems we encountered in previous chapters. In fact, although in some simplified situations it is possible to reduce the routing problem to a standard combinatorial optimization problem, in realistic settings the dynamics of the traffic, and therefore of the costs associated with network links, is such that it might even be impossible to give a formal definition of what an optimal solution is.

6.1.1 A Broad Classification of Routing Algorithms

Routing algorithms can be broadly classified as centralized versus distributed and as static versus adaptive.

In *centralized* algorithms, a main controller is responsible for updating all the node's routing tables and for making every routing decision. Centralized algorithms can be used only in particular cases and for small networks. In general, the delays necessary to gather information about the network status and to broadcast the decisions and the updates make them infeasible in practice. Moreover, centralized systems are not fault-tolerant: if the main controller does not work properly, all the network is affected. In contrast, in *distributed* routing, the computation of paths is shared among the network nodes, which exchange the necessary information. The distributed paradigm is currently used in the great majority of networks.

In *static* routing, the path taken by a data packet is determined only on the basis of its source and destination, without regard to the current network traffic. The path chosen is usually the minimum cost path according to some cost criterion, and can be changed only to account for faulty links or nodes. *Adaptive* routing is, in principle, more attractive, because it can adapt the routing policy to time and spatially varying traffic conditions. As a drawback, adaptive algorithms can cause oscillations in the selection of paths. This can cause circular paths, as well as large fluctuations in measured performance (Bertsekas & Gallager, 1992).

Another interesting way of looking at routing algorithms is from an optimization perspective. In this case the main choice is between optimal routing and shortest path routing.

Optimal routing has a network-wide perspective and its goal is to optimize a function of all individual link flows (usually this function is a sum of link costs assigned on the basis of average packet delays) (Bertsekas & Gallager, 1992).

Shortest-path routing has a source-destination pair perspective: there is no global cost function to optimize. Its objective is to determine the shortest path (minimum cost) between two nodes, where the link costs are computed (statically or adaptively) according to some statistical description of the traffic flow crossing the links. Considering the different content stored in each routing table, shortest-path algorithms can be further subdivided into two classes called distance-vector and link-state (Steenstrup, 1995).

Distance-vector algorithms make use of routing tables consisting of a set of triples of the form (*destination, estimated distance, and next hop*), defined for all the destinations in the network and for all the neighbor nodes of the considered switch. In this case, the required topologic information is represented by the list of identifiers of the reachable nodes. The average per node memory occupation is in the order of $O(\varphi \cdot n)$, where φ is the average connectivity degree (i.e., the average number of neighbor nodes considered over all the nodes) and n is the number of nodes in the network. The algorithm works in an iterative, asynchronous, and distributed way. The information that every node sends to its neighbors is the list of its last estimates of the distances (intended as costs) from itself to all the other nodes in the network. After receiving this information from a neighbor node j, the receiving node i updates its table of distance estimates overwriting the entry corresponding to node j with the received values. Routing decisions at node i are made choosing as the next hop node the one satisfying the expression $\arg\min_{j \in \mathcal{N}_i}\{d_{ij} + D_j\}$, where d_{ij} is the assigned cost to the link connecting node i with its neighbor j and D_j is the estimated shortest distance from node j to the destination. It can be shown that this algorithm converges in finite time to the shortest paths with respect to the used metric if no link cost changes after a given time (Bellman, 1958; Ford & Fulkerson, 1962; Bertsekas & Gallager, 1992); this algorithm is also known as distributed *Bellman-Ford*.

Link-state algorithms make use of routing tables containing much more information than that used in distance-vector algorithms. In fact, at the core of link-state algorithms there is a distributed and replicated database. This database is essentially a dynamic map of the whole network, describing the details of all its components and their current interconnections. Using this database as input, each node calculates its best paths using an appropriate algorithm such as Dijkstra's (Dijkstra, 1959), and then uses knowledge about these best paths to build the routing tables. The memory requirement for each node in this case is $\mathcal{O}(n^2)$. In the most common form of link-state algorithm, each node acts autonomously, broadcasting information about its link costs and states and computing shortest paths from itself to all the destinations on the basis of its local link cost estimates and of the estimates received from other nodes. Each routing information packet is broadcast to all the neighbor nodes which in turn send the packet to their neighbors, and so on. A distributed flooding mechanism (Bertsekas & Gallager, 1992) supervises this information transmission, trying to minimize the number of retransmissions.

It should be clear to the reader, from what was said in chapter 1, that ACO algorithms can easily be adapted to solve routing problems following the shortest-path/distance-vector paradigm.

6.1.2 The Communication Network Model

Before we can describe the AntNet algorithm, it is necessary to accurately define the problem we are going to consider. In particular, we need to define the network architecture and protocols, as well as the characteristics of the input data traffic. In turn, this also defines the characteristics of the network simulator that is used for the experiments.

In this chapter, we focus on irregular topology packet-switched data networks with an IP-like network layer (in the ISO-OSI terminology (Tanenbaum, 1996)) and a very simple transport layer. In particular, we focus on wide area networks (WANs), of which the Internet is a noteworthy instance. In WANs, *hierarchical* organization schemes are adopted. Roughly speaking, subnetworks are seen as single host nodes connected to interface nodes called gateways. Gateways perform fairly sophisticated network layer tasks, including routing. Groups of gateways, connected by an arbitrary topology, define logical areas. Inside each area, all the gateways are at the same hierarchical level and "flat" routing is performed among them. Areas communicate only by means of area border gateways. In this way, the computational complexity of the routing problem, as seen by each gateway, is much reduced, at the cost of an increase in the complexity of the design and management of the routing protocols.

The instances of the communication networks that we consider in the following can be mapped on directed weighted graphs with n processing/forwarding nodes. All the links between pairs of nodes are viewed as bit pipes characterized by a bandwidth (bit/s) and a transmission delay (s). Every node is of type store-and-forward and has a buffer space where the incoming and outgoing packets are stored. This buffer is a shared resource among all the queues attached to every incoming and outgoing link of the node. All the traveling packets are subdivided into two classes: data and routing packets. Additionally, there are two priority levels in queues. Usually, data packets are served in the low-priority queues, while routing packets are served in the high-priority queues. The workload is defined in terms of applications whose arrival rate is given by a probabilistic model. By application (or session, or connection in the following), we mean a process sending data packets from an origin node to a destination node. The number of packets to send, their sizes, and the intervals between them are assigned according to some defined stochastic process. We do not make any distinction among nodes, which act at the same time as hosts (session endpoints) and gateways/routers (forwarding elements). The adopted workload model incorporates a simple flow control mechanism implemented by using a fixed production window for the session's packets generation. The window determines the maximum number of data packets that can be waiting to be sent. Once sent, a packet is considered to be acknowledged. This means that the transport layer neither manages error control, nor packet sequencing, nor acknowledgments and retransmissions. (This choice, which is the same as in the "Simple_Traffic" model in the MaRS network simulator (Alaettinoğlu, Shankar, Dussa-Zieger, & Matta, 1992), can be seen as a very basic form of file transfer protocol (FTP).)

For each incoming packet, the node's routing component uses the information stored in the local routing table to choose the outgoing link to be used to forward the packet toward its destination node. When the link resources become available, they are reserved and the transfer is set up. The time it takes to move a packet from one node to a neighboring one depends on the packet size and on the link's transmission characteristics. If, on a packet's arrival, there is not enough buffer space to hold it, the packet is discarded. Otherwise, a service time is stochastically generated for the newly arrived packet. This time represents the delay between the packet arrival time and the time when it will be put in the buffer queue of the outgoing link the local routing component has selected for it.

Situations causing a temporary or steady alteration of the network topology or of its physical characteristics (link or node failure, adding or deleting of network components, and so on) are not taken into account in the discussed implementation, though it is easy to add them.

In order to run experiments with AntNet, a complete network simulator was developed in C++ by Gianni Di Caro (Di Caro, 2003; Di Caro & Dorigo, 1998c). It is a discrete event simulator using as its main data structure an event list, which holds the next future events. The simulation time is a continuous variable and is set by the currently scheduled event. The aim of the simulator is to closely mirror the essential features of the concurrent and distributed behavior of a generic communication network without sacrificing efficiency and flexibility in code development.

6.2 The AntNet Algorithm

AntNet, the routing algorithm we discuss in this chapter, is a direct extension of the Simple Ant Colony Optimization algorithm discussed in chapter 1. As will become clear in the following, AntNet is even closer to the real ants' behavior that inspired the development of the ACO metaheuristic than the ACO algorithms for $\mathcal{NP}$-hard problems that we discussed in previous chapters.

Informally, the AntNet algorithm and its main characteristics can be summarized as follows.

- At regular intervals, and concurrently with the data traffic, from each network node artificial ants are asynchronously launched toward destination nodes selected according to the traffic distribution [see equation (6.2)].
- Artificial ants act concurrently and independently, and communicate in an indirect way (i.e., stigmergically; see chapter 7, section 7.3), through the pheromones they read and write locally on the nodes.
- Each artificial ant searches for a minimum cost path joining its source and destination node.
- Each artificial ant moves step by step toward its destination node. At each intermediate node a greedy stochastic policy is applied to choose the next node to move to. The policy makes use of (1) node-local artificial pheromones, (2) node-local problem-dependent heuristic information, and (3) the ant's memory.
- While moving, the artificial ants collect information about the time length, the congestion status, and the node identifiers of the followed path.
- Once they have arrived at the destination, the artificial ants go back to their source nodes by moving along the same path as before but in the opposite direction.
- During this backward travel, node-local models of the network status and the pheromones stored on each visited node are modified by the artificial ants as a function of the path they followed and of its goodness.
- Once they have returned to their source node, the artificial ants are deleted from the system.

In the following subsections the above scheme is explained, all its components are described and discussed, and a more detailed description of the algorithm is given.

6.2.1 AntNet: Data Structures

In AntNet, artificial ants move on the construction graph $G_C = (C, L)$, with the constraint of never using the set of links that do not belong to the network graph (see also chapter 2, section 2.2.1). In practice, therefore, artificial ants move on the network graph.

Like all ACO algorithms, AntNet exploits artificial pheromone trails. These are maintained in an artificial pheromone matrix $\mathcal{T}_i$ associated with each node i of the data network. The elements τ_{ijd}'s of $\mathcal{T}_i$ indicate the learned desirability for an ant in node i and with destination d to move to node j. In AntNet pheromones have three indices because the considered problem consists of the solution of many, $n(n-1)/2$, minimum cost paths problems simultaneously. Therefore, an ant on a node i can in principle have any of the remaining $n-1$ nodes as destination. Hence the notation τ_{ijd}, in which different pheromones are associated with different destination nodes (this notation differs from the one used in chapter 3 in which pheromones do not correspond to specific destinations and are therefore denoted by τ_{ij}'s).

Another specificity of AntNet, shared with ACO algorithms within the hyper-cube framework (see chapter 3, section 3.4.3), is that τ_{ijd}'s are normalized to 1:

$$\sum_{j \in \mathcal{N}_i} \tau_{ijd} = 1, \quad d \in [1, n] \text{ and } \forall i,$$

where $\mathcal{N}_i$ is the set of neighbors of node i, and $n = |C|$.

Additionally, AntNet maintains at each node i a simple parametric statistical model $\mathcal{M}_i$ of the traffic situation over the network as seen by node i. This local model is used to evaluate the paths produced by the artificial ants. In fact, unlike the typical situation found in applications of ACO to $\mathcal{NP}$-hard problems, in network routing it is rather difficult to evaluate the quality of a path having as sole information the time it took for the artificial ant to traverse it: this is because the time it takes to go from a source to a destination node depends not only on the routing decisions but also on the network traffic. The model $\mathcal{M}_i(\mu_{id}, \sigma_{id}^2, \mathcal{W}_{id})$ is adaptive and described by the sample mean μ_{id} and the variance σ_{id}^2 computed over the trip times experienced by the artificial ants, and by a moving observation window $\mathcal{W}_{id}$ used to store the best value $W_{best_{id}}$ of the artificial ants' trip time. For each destination d in the network, the estimated mean and variance, μ_{id} and σ_{id}^2, give a representation of the expected time to go from node i to node d and of its stability. To compute these statistics AntNet uses the following exponential models:

$$\mu_{id} \leftarrow \mu_{id} + \varsigma(o_{i\to d} - \mu_{id}),$$
$$\sigma^2_{id} \leftarrow \sigma^2_{id} + \varsigma((o_{i\to d} - \mu_{id})^2 - \sigma^2_{id}), \tag{6.1}$$

where $o_{i\to d}$ is the new observed agent's trip time from node i to destination d. The factor ς (read: *varsigma*) weighs the number of most recent samples that will really affect the average. The weight of the k-th sample used to estimate the value of μ_{id} after j samplings, with $j > k$, is: $\varsigma(1-\varsigma)^{j-k}$. In this way, for example, if $\varsigma = 0.1$, approximately only the latest fifty observations will really influence the estimate, for $\varsigma = 0.05$, the latest 100, and so on. Therefore, the number of effective observations is approximately $5/\varsigma$.

As we said, $\mathcal{W}_{id}$ is used to store the value $W_{best_{id}}$ of the best ants' trip time from node i toward destination d as observed in the last w samples. The value $W_{best_{id}}$ represents a short-term memory expressing an estimate of the optimal time to go to node d from the current node. After each new sample, the length w of the window is incremented modulus w_{max}, where w_{max} is the maximum allowed size of the observation window and is set to $w_{max} = 5c/\varsigma$, with $c \leq 1$, so that, when $c = 1$, the value $W_{best_{id}}$ and the exponential estimates refer to the same set of observations.

In this way, the long-term exponential mean and the short-term windowing are referring to a comparable set of observations.

$\mathcal{T}$ and $\mathcal{M}$, illustrated in figure 6.1, can be seen as memories local to nodes capturing different aspects of the network dynamics. The model $\mathcal{M}$ maintains absolute distance/time estimates to all the nodes, while the pheromone matrix gives relative goodness measures for each link-destination pair under the current routing policy implemented over all the network.

6.2.2 AntNet: The Algorithm

AntNet is conveniently described in terms of two sets of artificial ants, called in the following *forward* and *backward ants*. Ants in each set possess the same structure, but they are differently situated in the environment; that is, they can sense different inputs and they can produce different, independent outputs. Ants communicate in an indirect way, according to the stigmergy paradigm, through the information they concurrently read and write on the network nodes they visit.

The AntNet algorithm, whose high-level description in pseudo-code is given in figure 6.2, can be described as being composed of two main phases: solution construction, and data structures update. These are described in the following.

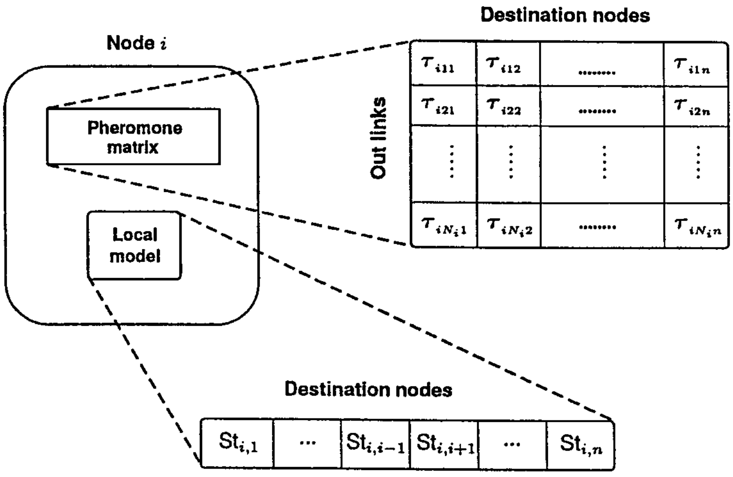


Figure 6.1
Data structures used by the artificial ants in AntNet for the case of a node i with $N_i = |\mathcal{N}_i|$ neighbors and a network with n nodes. The pheromone matrix $\mathcal{T}_i = [\tau_{ijd}]$ is isomorphic to the routing table used by the ants. The structure $\mathcal{M}_i = [\mathrm{St}_{i,d}]$, $d = 1, \dots n$, $d \neq i$, containing the statistics $\mu_{id}, \sigma^2_{id}, \mathcal{W}_{id}$ about the local traffic, plays the role of a local adaptive model for the expected delay toward each possible destination.

Solution Construction

At regular intervals Δt from every network node s, a forward ant $F_{s \to d}$ is launched toward a destination node d to discover a feasible, low-cost path to that node and to investigate the load status of the network along the path. Forward ants share the same queues as data packets, so that they experience the same traffic load. Destinations are locally selected according to the data traffic patterns generated by the local workload: if f_{sd} is a measure (in bits or in the number of packets) of the data flow $s \to d$, then the probability of creating at node s a forward ant with node d as destination is

$$p_{sd} = \frac{f_{sd}}{\sum_{i=1}^{n} f_{si}}. \tag{6.2}$$

In this way, ants adapt their exploration activity to the varying data traffic distribution.

While traveling toward their destination nodes, the forward ants keep memory of their paths and of the traffic conditions found. The identifier of every visited node i and the time elapsed since the launching time to arrive at this i-th node are stored in a memory stack $S_{s \to d}(i)$. The ant builds a path performing the following steps:

```
procedure AntNet($t$, $t_{end}$, $\Delta t$)
  input  $t$        % current time
  input  $t_{end}$  % time length of the simulation
  input  $\Delta t$ % time interval between ants generation

  foreach $i \in C$ do                  % concurrent activity over the network
    $\mathcal{M} \leftarrow$ InitLocalTrafficModel
    $\mathcal{T} \leftarrow$ InitNodeRoutingTable
    while $t \leq t_{end}$ do
      in_parallel                       % concurrent activity on each node
        if $(t \bmod \Delta t) = 0$ then
          $destination \leftarrow$ SelectDestination($traffic\_distribution\_at\_source$)
          LaunchForwardAnt($source$, $destination$)
        end-if
        foreach (ActiveForwardAnt[$source$, $current$, $destination$]) do
          while ($current \neq destination$) do
            $next\_hop \leftarrow$ SelectLink($current$, $destination$, $link\_queues$, $\mathcal{T}$)
            PutAntOnLinkQueue($current$, $next\_hop$)
            WaitOnDataLinkQueue($current$, $next\_hop$)
            CrossLink($current$, $next\_hop$)
            Memorize($next\_hop$, $elapsed\_time$)
            $current \leftarrow next\_hop$
          end-while
          LaunchBackwardAnt($destination$, $source$, $memory\_data$)
        end-foreach
        foreach (ActiveBackwardAnt[$source$, $current$, $destination$]) do
          while ($current \neq destination$) do
            $next\_hop \leftarrow$ PopMemory
            WaitOnHighPriorityLinkQueue($current$, $next\_hop$)
            CrossLink($current$, $next\_hop$)
            $from \leftarrow current$
            $current \leftarrow next\_hop$
            UpdateLocalTrafficModel($\mathcal{M}$, $current$, $from$, $source$, $memory\_data$)
            $r \leftarrow$ GetNewPheromone($\mathcal{M}$, $current$, $from$, $source$, $memory\_data$)
            UpdateLocalRoutingTable($\mathcal{T}$, $current$, $source$, $r$)
          end-while
        end-foreach
      end-in_parallel
    end-while
  end-foreach
end-procedure
```

1. At each node i, each forward ant headed toward a destination d selects the node j to move to, choosing among the neighbors it did not already visit, or over all the neighbors in case all of them had previously been visited. The neighbor j is selected with a probability P_{ijd} computed as the normalized sum of the pheromone τ_{ijd} with a heuristic value η_{ij} taking into account the state (the length) of the j-th link queue of the current node i:

$$P_{ijd} = \frac{\tau_{ijd} + \alpha\eta_{ij}}{1 + \alpha(|\mathcal{N}_i| - 1)}. \tag{6.3}$$

The heuristic value η_{ij} is a $[0, 1]$ normalized value function of the length q_{ij} (in bits waiting to be sent) of the queue on the link connecting the node i with its neighbor j:

$$\eta_{ij} = 1 - \frac{q_{ij}}{\sum_{l=1}^{|\mathcal{N}_i|} q_{il}}. \tag{6.4}$$

The value of α weighs the importance of the heuristic value with respect to the pheromone values stored in the pheromone matrix $\mathcal{T}$ (similar to what is done in the ANTS algorithm; see chapter 3, section 3.4.2). The value η_{ij} reflects the instantaneous state of the node's queues and, assuming that the queue's consuming process is almost stationary or slowly varying, η_{ij} gives a quantitative measure associated with the queue waiting time. The pheromone values, on the other hand, are the outcome of a continual learning process and capture both the current and the past status of the whole network as seen by the local node. Correcting these values with the values of η allows the system to be more "reactive," and at the same time it avoids following all the network fluctuations. An ant's decisions are therefore taken on the basis of a combination of a long-term learning process and an instantaneous heuristic prediction.

2. If a cycle is detected, that is, if an ant returns to an already visited node, the cycle's nodes are removed and all the memory about them is deleted. If the cycle lasted longer than the lifetime of the ant before entering the cycle, that is, if the cycle is greater than half the ant's age, the ant is deleted. In fact, in this case the agent wasted a lot of time, probably because of a wrong sequence of decisions and not because of congestion states. Therefore, the agent is carrying an old and misleading

Figure 6.2
AntNet's high-level description in pseudo-code. All the described actions take place in a completely distributed and concurrent way over the network nodes (while, in the text, AntNet has been described from an individual ant's perspective). All the constructs at the same level of indentation inside the context of the statement `in_parallel` are executed concurrently. The processes of data generation and forwarding are not described, but they can be thought as acting concurrently with the ants.

memory of the network's state and it could be counterproductive to use it to update the pheromone trails (see below).

3. When the destination node d is reached, the agent $F_{s \to d}$ generates another agent (backward ant) $B_{d \to s}$, transfers to it all of its memory, and is deleted. A forward ant is also deleted if its lifetime becomes greater than a value *max_life* before it reaches its destination node, where *max_life* is a parameter of the algorithm.

4. The backward ant takes the same path as that of its corresponding forward ant, but in the opposite direction. Backward ants do not share the same link queues as data packets; they use higher-priority queues reserved for routing packets, because their task is to quickly propagate to the pheromone matrices the information accumulated by the forward ants.

Data Structures Update
Arriving at a node i coming from a neighbor node, the backward ant updates the two main data structures of the node, the local model of the traffic $\mathcal{M}_i$ and the pheromone matrix $\mathcal{T}_i$, for all the entries corresponding to the (forward ant) destination node d. With some precautions, updates are performed also on the entries corresponding to every node $d' \in S_{i \to d}$, $d' \neq d$ on the "subpaths" followed by ant $F_{s \to d}$ after visiting the current node i. In fact, if the elapsed trip time of a subpath is statistically "good" (i.e., less than $\mu_{id} + I(\mu_{id}, \sigma_{id})$, where I is an estimate of a confidence interval for μ_{id}), then the time value is used to update the corresponding statistics and the pheromone matrix. On the contrary, trip times of subpaths that are not deemed good, in the same statistical sense as defined above, are not used because they might give a wrong estimate of the time to go toward the subdestination node. In fact, all the forward ant routing decisions were made only as a function of the destination node. In this perspective, subpaths are side effects, and they are potentially suboptimal because of local variations in the traffic load. Obviously, in the case of a good subpath, it can be used: the ant discovered, at zero cost, an additional good route. In the following, we describe the way $\mathcal{M}_i$ and $\mathcal{T}_i$ are updated with respect to a generic "destination" node $d' \in S_{i \to d}$. A simple example of the way AntNet's ants update $\mathcal{M}_i$ and $\mathcal{T}_i$ is given in figure 6.3.

- $\mathcal{M}_i$ is updated with the values stored in the backward ant's memory. The time elapsed to arrive (for the forward ant) to the destination node d' starting from the current node is used to update, according to equation (6.1), the mean and variance estimates, $\mu_{id'}$ and $\sigma^2_{id'}$, as well as the best value over the observation window $\mathcal{W}_{id'}$. In this way, a parametric model of the traveling time from node i to destination d' is maintained. The mean value of this time and its dispersion can vary strongly, depending on the traffic conditions: a poor time (path) under low traffic load can be

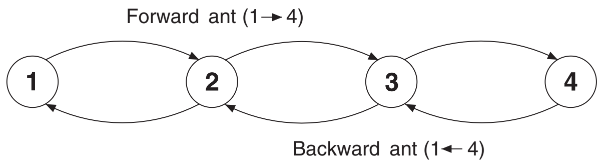


Figure 6.3
Example of the way AntNet's ants update node data structures. The forward ant, $F_{1\to4}$, moves along the path $1 \to 2 \to 3 \to 4$ and, arrived at node 4, launches the backward ant $B_{4\to1}$ which travels in the opposite direction. At each node i, $i = 3, \ldots, 1$, the backward ant uses its memory contents $S_{1\to4}(i)$ to update the values for $\mathcal{M}_i(\mu_{i4}, \sigma^2_{i4}, \mathcal{W}_{i4})$, and, in case of good subpaths, to update also the values for $\mathcal{M}_i(\mu_{id'}, \sigma^2_{id'}, \mathcal{W}_{id'})$, $d' = i+1, \ldots, 3$. At the same time, the pheromone matrix is updated by incrementing the pheromone τ_{ij4}, $j = i+1$, of the last node j the ant $B_{4\to1}$ came from, and decrementing the values of the pheromones of the other neighbors (not shown in the figure). The increment will be a function of the trip time experienced by the forward ant going from node i to destination node 4. As for $\mathcal{M}_i$, if the trip time associated by the forward ant with the subpaths to the other nodes $d' = i+1, \ldots, 3$ is statistically good, then also the corresponding pheromone matrix entries are updated. Adapted from Di Caro & Dorigo (1998c).

a very good one under heavy traffic load. The statistical model has to be able to capture this variability and to follow in a robust way the fluctuations of the traffic. This model plays a critical role in the pheromone matrix updating process, as explained in the following.

- The pheromone matrix $\mathcal{T}_i$ is updated by incrementing the pheromone $\tau_{ifd'}$ (i.e., the pheromone suggesting to choose neighbor f when destination is d') and decrementing, by normalization, the other pheromones $\tau_{ijd'}$, $j \in \mathcal{N}_i$, $j \neq f$. The way the pheromones are updated depends on a measure of goodness associated with the trip time $T_{i\to d'}$ experienced by the forward ant, and is given below. This time represents the only available explicit feedback signal to score paths. It gives a clear indication about the goodness of the followed path because it is proportional to its length from a physical point of view (number of hops, transmission capacity of the used links, processing speed of the crossed nodes) and from a traffic congestion point of view (because the forward ants share the same queues as data packets). Note that the time measure T cannot be associated with an exact error measure, given that the "optimal" trip times are not known, because they depend on the whole network load status. In fact, when the network is in a congested state, all the trip times will score poorly with respect to the times observed in low load situations. Nevertheless, a path with a high trip time should be scored as a good path if its trip time is significantly lower than the other trip times observed in the same congested situation. Therefore, T can only be used as a reinforcement signal. This gives rise to a credit assignment problem typical of the reinforcement learning field (Bertsekas & Tsitsiklis, 1996; Kaelbling, Littman, & Moore, 1996).

The reinforcement $r \equiv r(T, \mathcal{M}_i)$, $0 < r \leq 1$, is used to update the pheromones. It is computed by taking into account some average of the values observed so far and of their dispersion to score the goodness of the trip time T, such that the smaller the T, the higher the r (the exact definition of r is discussed in the next subsection). The value r is used by the backward ant $B_{d \to s}$ moving from node f to node i to increase the pheromone values $\tau_{ifd'}$. The pheromone $\tau_{ifd'}$ is increased by r as follows:

$$\tau_{ifd'} \leftarrow \tau_{ifd'} + r \cdot (1 - \tau_{ifd'}). \tag{6.5}$$

In this way, given a same value r, small pheromone values are increased proportionally more than large pheromone values, favoring in this way a quick exploitation of new, and good, discovered paths.

Pheromones $\tau_{ijd'}$ for destination d' of the other neighboring nodes j, $j \in \mathcal{N}_i$, $j \neq f$, evaporate implicitly by normalization. That is, their values are reduced so that the sum of pheromones on links exiting from node i will remain 1:

$$\tau_{ijd'} \leftarrow \tau_{ijd'} - r \cdot \tau_{ijd'}, \quad j \in \mathcal{N}_i,\ j \neq f. \tag{6.6}$$

It is important to remark that every discovered path increases its selection probability. In this way, not only does the (explicit) assigned value r play a role but also the (implicit) ant's arrival rate. In this respect, AntNet is closer to real ants' behavior than the other ACO algorithms for $\mathcal{NP}$-hard problems we have studied in the previous chapters: in fact, it exploits the *differential path length effect* described in chapter 1. This strategy is based on trusting paths that receive either high reinforcements, independent of their frequency, or low and frequent reinforcements. In fact, for any traffic load condition, a path receives one or more high reinforcements only if it is much better than previously explored paths. On the other hand, during a transient phase after a sudden increase in network load all paths will likely have high traversing times with respect to those learned by the model $\mathcal{M}$ in the preceding, low-congestion, situation. Therefore, in this case good paths can only be differentiated by the frequency of ants' arrivals. Assigning always a positive, but low, reinforcement value in the case of paths with high traversal time allows the implementation of the above mechanism based on the frequency of the reinforcements, while, at the same time, it avoids giving excessive credit to paths with high traversal time due to their poor quality.

6.2.3 How to Evaluate the Quality of an Ant's Trip

The value r is a critical quantity that has to be assigned after considering three main aspects: (1) paths should receive an increment in their selection probability proportional to their goodness, (2) the goodness is a relative measure, which depends on the traffic conditions that can be estimated by means of the models $\mathcal{M}_i$, and (3) it is important not to follow all the traffic fluctuations. This last aspect is particularly important. Uncontrolled oscillations in the routing tables are one of the main problems in shortest-path routing (Wang & Crowcroft, 1992). It is very important to be able to set the best trade-off between stability and adaptivity.

Several ways to assign the r values, trying to take into account the above three requirements, have been investigated:

- The simplest way is to set $r = constant$: independently of the ant's "experiment outcomes," the discovered paths are all rewarded in the same way. In this simple but meaningful case the core of the algorithm is based on the capability of "real" ants to discover shortest paths via stigmergic communication mediated by pheromone trails. In other words, what is at work is the differential path length effect: ants traveling along faster paths will arrive at a higher rate than other ants, hence their paths will receive a higher cumulative reward. The obvious problem with this approach lies in the fact that, although ants following longer paths arrive delayed, they will nevertheless have the same effect on the pheromone matrices as the ants that followed shorter paths.
- A more elaborate approach is to define r as a function of the ant's trip time T and of the parameters of the local statistical model $\mathcal{M}_i$. The following functional form gave good results, and was used in the experiments reported later in this chapter:

$$r = c_1\left(\frac{W_{best}}{T}\right) + c_2\left(\frac{I_{sup} - I_{inf}}{(I_{sup} - I_{inf}) + (T - I_{inf})}\right). \tag{6.7}$$

I_{sup} and I_{inf} are estimates of the limits of an approximate confidence interval for μ. I_{inf} is set to W_{best}, while $I_{sup} = \mu + z(\sigma/\sqrt{w})$, with $z = 1/\sqrt{(1-v)}$ where v gives the selected confidence level. This expression is obtained using the Tchebycheff inequality that allows the definition of a confidence interval for a random variable following any distribution (Papoulis, 2001). Although usually, for specific probability densities, the Tchebycheff bound is not very tight, here its use is justified by the fact that only a raw estimate of the confidence interval is needed and that in this way there is no need to make any assumption on the distribution of μ.

The first term in equation (6.7) simply evaluates the ratio between the best trip time observed over the current observation window and the current trip time. The second term evaluates how far the value T is from I_{inf} in relation to the extension of the confidence interval, that is, considering the stability in the latest trip times. Note that the denominator of this term could go to zero, when $T = I_{sup} = I_{inf}$. In this case the whole term is set to zero. The coefficients c_1 and c_2 are parameters which weigh the importance of each term.

The value r obtained from equation (6.7) is finally transformed by means of a squash function $s(x)$:

$$r = \frac{s(r)}{s(1)}, \tag{6.8}$$

where

$$s(x) = \left(1 + exp\left(\frac{a}{x|\mathcal{N}_i|}\right)\right)^{-1}, \quad x \in (0,1],\ a \in R^+. \tag{6.9}$$

Squashing the r-values allows the system to be more sensitive in rewarding good (high) values of r, while having the tendency to saturate the rewards for bad (near to zero) r-values: the scale is compressed for lower values and expanded in the upper part. In such a way an emphasis is put on good results.

6.3 The Experimental Settings

In this section we describe the test bed used to compare AntNet with some of the best-known routing algorithms. Note that, because the functioning of a data network is governed by many components which may interact in nonlinear and unpredictable ways, the choice of a meaningful test bed is not an easy task: the approach followed is to define a limited set of classes of tunable components. These are: the topology and the physical properties of the network, the traffic patterns, the metrics chosen for performance evaluation, the competing routing algorithms chosen, and their parameter values. In the following, for each class the choices are explained.

6.3.1 Topology and Physical Properties of the Net

The experiments presented in section 6.4 were run on models based on two real-world network instances: the US National Science Foundation network, NSFnet, and the Japanese NTT company backbone, NTTnet.

- *NSFnet* is the old USA T1 backbone (1987). NSFnet is a WAN composed of fourteen nodes and twenty-one bidirectional links with a bandwidth of 1.5 Mbit/s. Its topology is shown in figure 6.4. Propagation delays range from 4 to 20 ms. NSFnet is a well-balanced network, where a network is said to be well-balanced if the distribution of the shortest paths between all the pairs of nodes has a small variance.

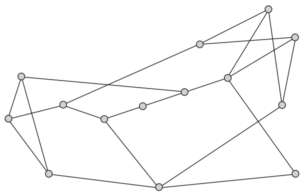


Figure 6.4
NSFnet. Each arc in the graph represents a pair of directed links. Link bandwidth is 1.5 Mbit/s; propagation delays range from 4 to 20 ms.

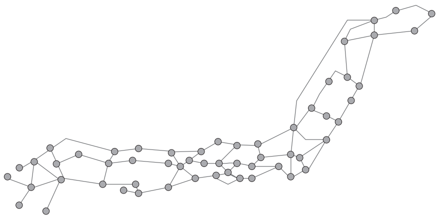


Figure 6.5
NTTnet. Each arc in the graph represents a pair of directed links. Link bandwidth is 6 Mbit/s, propagation delays range from 1 to 5 ms.

- *NTTnet* is a network modeled on the NTT (Nippon Telephone and Telegraph company) fiberoptic corporate backbone at the end of the '90s. It is a 57-node, 162 bidirectional links network. Link bandwidth is of 6 Mbit/s, while propagation delays range from 1 to 5 ms. Its topology is shown in figure 6.5. NTTnet is not a well-balanced network.

All the networks are simulated with link-fault and node-fault probabilities set to zero, local node buffers of 1 Gbit capacity, and data packet maximum time to live (TTL) set to 15 seconds.

6.3.2 Traffic Patterns

Traffic is defined in terms of open sessions between pairs of different nodes. Traffic patterns can show a huge variety of forms, depending on the characteristics of each session and on their distribution from geographic and temporal points of view. Each session is characterized by the number of transmitted packets, and by their size and interarrival time distributions. Sessions over a network can be characterized by their interarrival time distribution and by their geographic distribution. The latter is controlled by the probability assigned to each node to be selected as a session start or endpoint.

In the experiments three basic patterns for the temporal distribution of the sessions and three for their spatial distribution were considered.

Temporal Distributions

- *Poisson* (P): for each node a Poisson process regulates the arrival of new sessions (i.e., session interarrival times follow a negative exponential distribution).
- *Fixed* (F): at the beginning of the simulation, for each node a fixed number of one-to-all sessions is set up and left constant for the whole simulation.
- *Temporary* (TMPHS): a temporary, heavy-load traffic condition is generated, turning on some nodes that act like hot spots (see below).

Spatial Distributions

- *Uniform* (U): the assigned temporal characteristics for session arrivals are set to be identical in all the network nodes.
- *Random* (R): the assigned temporal characteristics for session arrivals are set randomly over the network nodes.
- *Hot spots* (HS): some nodes behave as hot spots, concentrating a high rate of input/output traffic. A fixed number of sessions are opened from the hot spots to all the other nodes.

General traffic patterns are obtained combining the above temporal and spatial characteristics. Therefore, for example, UP traffic means that on each node an identical Poisson process regulates the arrival of new sessions, while in the RP case the characteristics of the Poisson process are different for each node, and UP-HS means that a hot spots traffic model is superimposed on a UP traffic.

The bit streams generated by each session were chosen to have a time-varying bit rate (called *generic variable bit rate*, GVBR, in the following). The term GVBR is a broad generalization of the term *varying bit rate*, VBR, normally used to designate a bit stream with a variable bit rate but with known average characteristics and expected/admitted fluctuations. Here, a GVBR session generates packets whose sizes and interarrival times are variable and follow a negative exponential distribution. The information about these characteristics is never directly used by the routing algorithms, as in IP-based networks.

The values used in the experiments to shape traffic patterns are "reasonable" values for session generations and data packet production, taking into consideration network usage and computing power at the time the experiments were carried out (Di Caro & Dorigo, 1998c). The mean of the packet size distribution was set to 4096 bits in all the experiments. Basic temporal and spatial distributions are chosen to be representative of a wide class of possible situations that can be arbitrarily composed to generate a meaningful subset of real traffic patterns.

6.3.3 Metrics for Performance Evaluation

The metrics used for performance evaluation are *throughput* (correctly delivered bit/s) and *delay distribution* for data packets (s). These are the standard metrics for performance evaluation, when considering only sessions with equal costs, benefits, and priority and without the possibility of requests for special services like real time. Simulation results for throughput are reported as average values without an associated measure of variance. The intertrial variability is in fact always very low, within a few percentage points of the average value. Simulation results concerning packet delays are reported either using the whole empirical distribution or the 90th percentile, which allows comparison of algorithms on the basis of the upper value of delay they were able to keep 90% of the correctly delivered packets. In fact, packet delays can be spread over a wide range of values. This is an intrinsic characteristic of data networks: packet delays can range from very low values for sessions open between adjacent nodes connected by fast links, to much higher values in the case of sessions involving nodes very far apart connected by many slow links. Because of this, very often the empirical distribution of packet delays cannot be meaningfully parameterized in terms of mean and variance, and the 90th percentile statistic, or still better, the whole empirical distribution, is much more meaningful.

6.3.4 Competing Routing Algorithms and Their Parameters

AntNet performance was compared with state-of-the-art routing algorithms taken from the telecommunications and machine learning literature. The algorithms were reimplemented to make them as efficient as possible. They belong to the various possible combinations of static and adaptive, distance-vector, and link-state classes, and are listed below:

OSPF (static, link-state) is an implementation of the current Interior Gateway Protocol (IGP) of Internet (Moy, 1998). It is essentially a static shortest path algorithm.

SPF (adaptive, link-state) is the prototype of link-state algorithms with a dynamic metric for link cost evaluations. A similar algorithm was implemented in the second version of ARPANET (McQuillan, Richer, & Rosen, 1980) and in its successive revisions (Khanna & Zinky, 1989).

Table 6.1
Routing packets size for the implemented algorithms (except for the Daemon algorithm, which does not generate routing packets)

	AntNet	OSPF & SPF	BF	Q-R & PQ-R
Packet size (byte)	$24 + 8H$	$64 + 8\|\mathcal{N}_i\|$	$24 + 12n$	12

H is the incremental number of hops made by the forward ant, $|\mathcal{N}_i|$ is the number of neighbors of node i, and n is the number of network nodes. The values assigned to these parameters are either the same as used in previous simulation works (Alaettinoğlu et al., 1992) or were chosen on the basis of heuristic evaluations (e.g., the size of forward ants was set to be the same size as that of a BF packet plus 8 bytes for each hop to store the information about the node address and the elapsed time).

BF (adaptive, distance-vector) is the asynchronous distributed Bellman-Ford algorithm with dynamic metrics (Bertsekas & Gallager, 1992; Shankar et al., 1992).

Q-R (adaptive, distance-vector) is the Q-Routing algorithm proposed by Boyan & Littman (1994) (an online asynchronous version of the Bellman-Ford algorithm).

PQ-R (adaptive, distance-vector) is the Predictive Q-Routing algorithm of Choi & Yeung (1996).

Daemon (adaptive, optimal routing) is an approximation of an ideal algorithm defining an empirical upper bound on the achievable performance. The algorithm exploits a "daemon" able to read in every instant the state of all the queues in the network and then calculates instantaneous "real" costs for all the links and assigns paths on the basis of a network-wide shortest-paths recalculation for every packet hop.

All the algorithms used have a collection of parameters to be set. Common parameters are routing packet size and elaboration time. Settings for these parameters are shown in table 6.1.

Concerning the other main parameters, specific for each algorithm, for the AntNet competitors either the best settings available in the literature were used or the parameters were tuned as much as possible to obtain better results. For OSPF, SPF, and BF, the length of the time interval between consecutive routing information broadcasts and the length of the time window to average link costs are the same, and they are set to 0.8 or 3.0 seconds, depending on the experiment for SPF and BF, and to 30 seconds for OSPF. For Q-R and PQ-R the transmission of routing information is totally data-driven. The learning and adaptation rate used were the same as those used by the algorithms' authors (Boyan & Littman, 1994; Choi & Yeung, 1996).

Concerning AntNet, the algorithm is very robust to internal parameter settings. The parameter set was not fine-tuned and the same set of values was used in all the

Box 6.1
Parameter Settings for AntNet

In this box we report "good" values for AntNet's parameters. "Good" means that the value of these parameters was not optimized experimentally, so that it is to be expected that AntNet performance can be slightly increased by their careful optimization. Nevertheless, AntNet's performance was found to be rather robust with respect to limited variations in these parameter values.

- $\varsigma = 0.005$: exponential mean coefficient found in equation (6.1).
- $\Delta t = 0.3$ second: time interval between two consecutive ant generations.
- $\alpha = 0.45$: relative weight of heuristic information with respect to pheromones, found in equation (6.3). In all the experiments that were run it was observed that the use of the heuristic value is a very effective mechanism: depending on the characteristics of the problem, the best value to assign to the weight α can vary, but if α ranges between 0.2 and 0.5, performance doesn't change appreciably. For lower values, the effect of η_{ij} is vanishing, while for higher values the resulting routing tables oscillate and, in both cases, performance degrades.
- $max\_life = 15$: number of hops after which an ant is removed from the system.
- $w_{max} = 5(c/\varsigma)$, with $c = 0.3$: max length of the observation windows (see section 6.2.1).
- $c_1 = 0.7$, $c_2 = 0.3$: constants found in equation (6.7), to compute the value r used to update pheromones. Experiments have shown that c_2 should not be too big (i.e., smaller than 0.35), otherwise performance starts to degrade appreciably. The behavior of the algorithm is quite stable for c_2 values in the range 0.15 to 0.35, but setting c_2 below 0.15 slightly degrades performance.
- $I_{inf} = W_{best}$, $I_{sup} = \mu + z(\sigma/\sqrt{w})$, with $z = 1.7$: values found in equation (6.7).
- $a = 10$: constant found in equation (6.9).

different experiments presented in the next section. The settings for all parameters used by AntNet are summarized in box 6.1.

6.4 Results

In this section we compare AntNet with the competing routing algorithms described in section 6.3.4. The performance of the algorithms was studied for increasing traffic load and for temporary saturation conditions. In the experiments reported here, the saturating input traffic, whose level is a function of the routing algorithm used, was determined using AntNet as routing algorithm.

All reported data are averaged over ten trials lasting 1000 virtual seconds of simulation time, which was found to be a time interval long enough to make effects due to transients negligible and to get enough statistical data to evaluate the behavior of the routing algorithm. Before being fed with data traffic, the algorithms are given 500 preliminary simulation seconds with no data traffic to build initial routing tables. In this way, each algorithm builds the routing tables according to its own "vision" about minimum cost paths in relation to the physical characteristics of the network.

Parameter values for traffic characteristics are given in the figure captions with the following meaning: *MSIA* is the mean of the session interarrival time distribution for the Poisson (P) case, *MPIA* is the mean of the packet interarrival time distribution, *HS* is the number of hot-spot nodes, and *MPIA-HS* is the equivalent of MPIA for the hot-spot sessions. As we said (see section 6.3.2), the shape of the session bit streams is of the GVBR type.

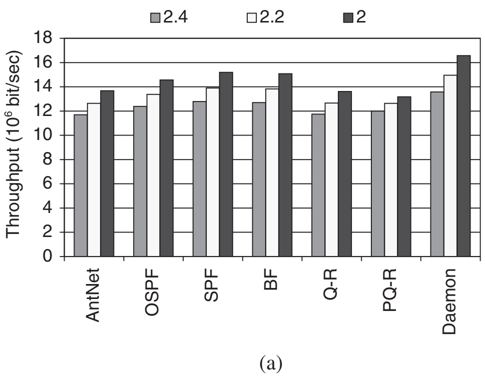
It should be noted that when using AntNet, data packets are routed in a probabilistic way. This has been observed to improve AntNet performance, in some cases even by 30% to 40%, which means that the way the routing tables are built in AntNet is well matched with a probabilistic distribution of the data packets over all the good paths. Data packets are prevented from choosing links with very low probability by remapping the elements of the routing table P by means of a power function $f(x) = x^{\delta}$, $\delta > 1$, which emphasizes high probability values and reduces lower ones. This value was set to $\delta = 1.2$ in the experiments. Differently, the use of probabilistic data routing was found not to improve the performance of the algorithms used for comparison. Therefore, in all the other algorithms the routing of data was done deterministically by choosing at each hop the best neighbor among those indicated in the routing table.

Results for *throughput* and *packet delays* for all the considered network topologies are described in the two following subsections. Results concerning the *network resources utilization* are reported in section 6.4.3.

6.4.1 NSFnet

Experiments on NSFnet were run using UP, RP, UP-HS, and TMPHS-UP traffic patterns. In all the cases considered, differences in throughput were found to be of minor importance with respect to those shown by packet delays. For each of the UP, RP, and UP-HS cases, three distinct groups of ten trial experiments were run, gradually increasing the generated workload (in terms of reducing the session interarrival time). This amounts, as explained above, to studying the behavior of the algorithms when moving the traffic load toward a saturation region.

In the UP case, differences in throughput (figure 6.6a) were found to be small: the best performing algorithms were BF and SPF, which attained performances only about 10% inferior to that of Daemon and of the same amount better than those of AntNet, Q-R, and PQ-R, while OSPF behaved slightly better than the last-named. Concerning delays (figure 6.6b), the results were rather different: OSPF, Q-R, and PQ-R performed poorly, while BF and SPF had a performance on the order of 50% worse than that obtained by AntNet and 65% worse than Daemon.



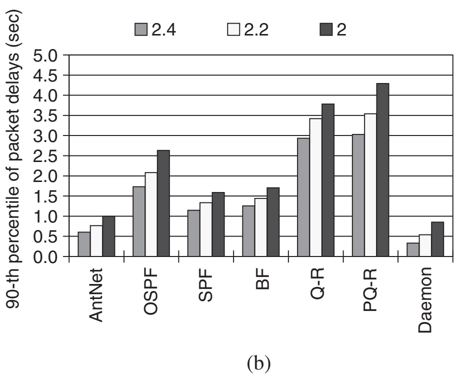


Figure 6.6
NSFnet: comparison of algorithms for increasing load for UP traffic. The load is increased reducing the MSIA value from 2.4 to 2.0 seconds (MPIA = 0.005 second). Statistics are computed over ten trials: (a) average throughput; (b) 90th percentile of the packet delays empirical distribution.

In the RP case (figure 6.7a), throughputs generated by AntNet, SPF, and BF were very similar, although AntNet presented a slightly better performance. OSPF and PQ-R behaved only slightly worse, while Q-R was the worst algorithm. Daemon was able to obtain only slightly better results than AntNet. Again, looking at packet delay results (figure 6.7b) OSPF, Q-R, and PQ-R performed very badly, while SPF showed results a bit better than those of BF but approximately 40% worse than those of AntNet. Daemon was in this case far better, which indicates that the test bed was very difficult.

For the case of UP-HS load, throughputs (figure 6.8a) for AntNet, SPF, BF, Q-R, and Daemon were found to be very similar, while OSPF and PQ-R gave much worse results. Again (figure 6.8b), packet delay results for OSPF, Q-R and PQ-R were much worse than those of the other algorithms (they were so much worse that they did not fit the scale chosen to highlight the differences between the other algorithms). AntNet was once again the best-performing algorithm (except, as usual, for Daemon). In this case, differences with SPF were found to be around 20%, and about 40% with respect to BF. Daemon performed about 50% better than AntNet and scaled much better than AntNet, which, again, indicates that the test bed was rather difficult.

The last graph for NSFnet shows how the algorithms behave in the case of a TMPHS-UP situation (figure 6.9). At time $t = 400$ four hot spots were turned on and superimposed on the existing light UP traffic. The transient was kept on for 120 seconds. In this case, only one, typical, situation is reported in detail to show how the different algorithms reacted. Reported values are the "instantaneous" values for

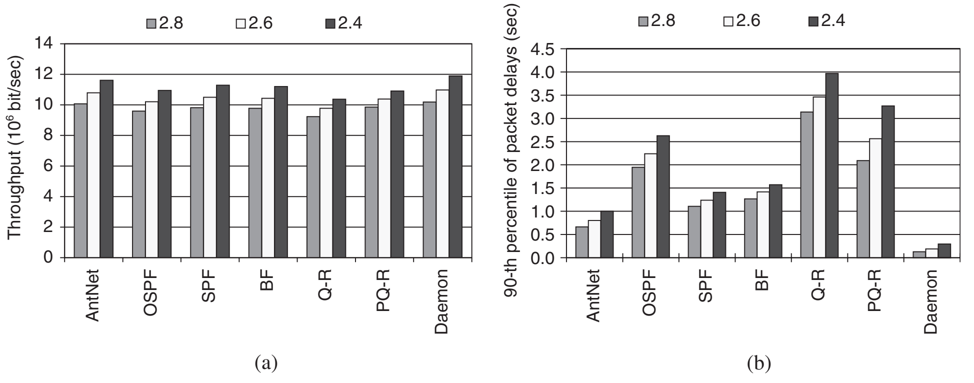


Figure 6.7
NSFnet: comparison of algorithms for increasing load for RP traffic. The load is increased reducing the MSIA value from 2.8 to 2.4 seconds (MPIA = 0.005 second). Statistics are computed over ten trials: (a) average throughput; (b) 90th percentile of the packet delays empirical distribution.

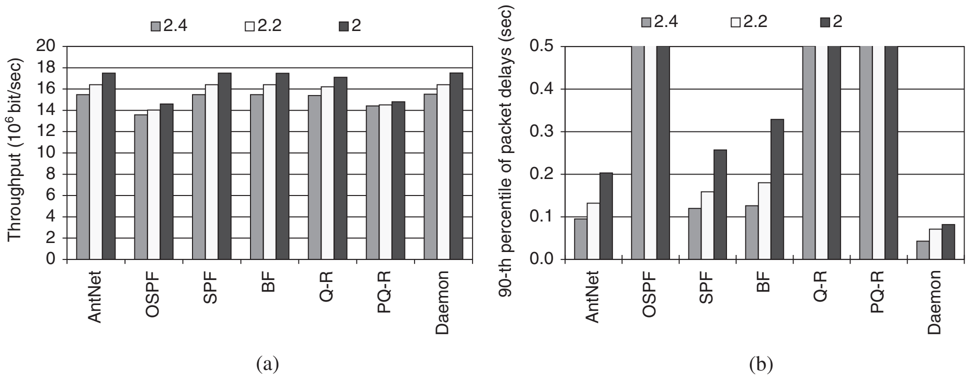


Figure 6.8
NSFnet: comparison of algorithms for increasing load for UP-HS traffic. The load is increased reducing the MSIA value from 2.4 to 2.0 seconds (MPIA = 0.3 second, HS = 4, MPIA-HS = 0.04 second). Statistics are computed over ten trials: (a) average throughput; (b) 90th percentile of the packet delays empirical distribution.

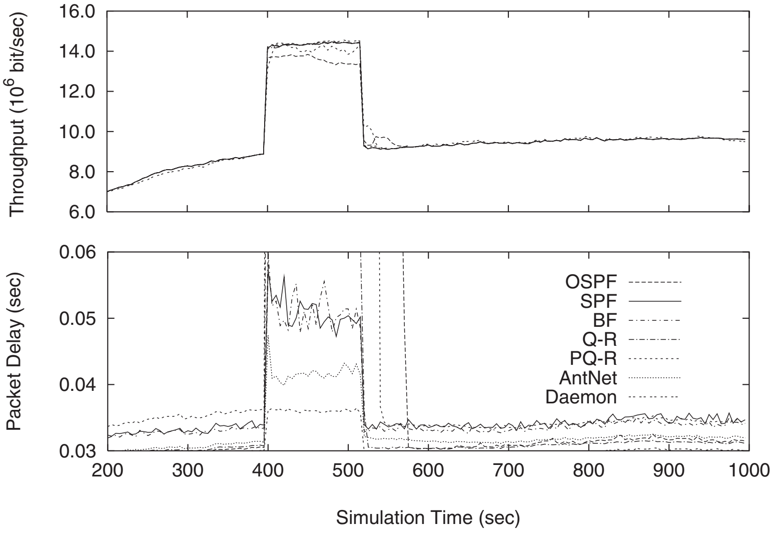


Figure 6.9
NSFnet: comparison of algorithms for transient saturation conditions with TMPHS-UP traffic (MSIA = 3.0 seconds, MPIA = 0.3 second, HS = 4, MPIA-HS = 0.04). Statistics are computed over ten trials: (up) average throughput; (down) packet delays averaged over 5-second moving windows. Reprinted by permission from Di Caro & Dorigo (1998c), © Morgan Kaufmann Publisher.

throughput and packet delays computed as the average over 5-second moving windows. All algorithms had a similar very good performance as far as throughput is concerned, except for OSPF and PQ-R, which lost a small percentage of the packets during the transitory period. The graph of packet delays confirms previous results: SPF and BF have a similar behavior, about 20% worse than AntNet and 45% worse than Daemon. The other three algorithms show a big out-of-scale jump not being able to properly dump the sudden load increase.

6.4.2 NTTnet

The same set of experiments run on the NSFnet was repeated on the NTTnet. In this case the results are even sharper than those obtained with NSFnet: AntNet performance is much better than that of all its competitors.

For the UP, RP, and UP-HS cases, differences in throughput are not significant (figures 6.10a, 6.11a, and 6.12a). All the algorithms, except the OSPF, practically behave in the same way as the Daemon algorithm. Concerning packet delays (figures 6.10b, 6.11b, and 6.12b), differences between AntNet and each of its competitors are

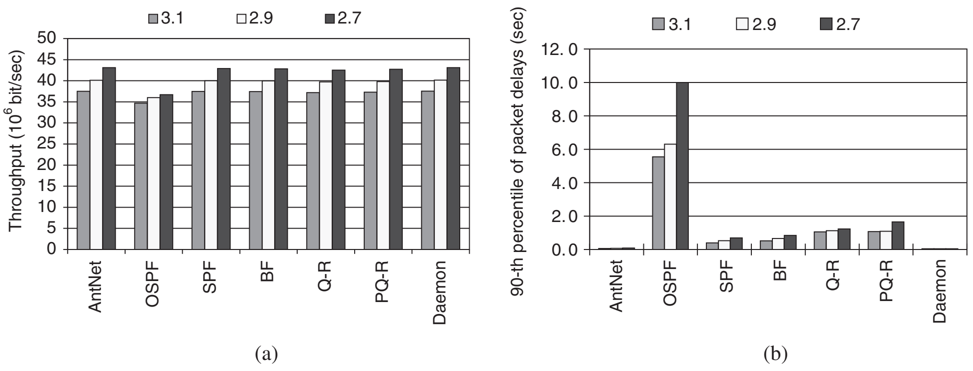


Figure 6.10
NTTnet: comparison of algorithms for increasing load for UP traffic. The load is increased reducing the MSIA value from 3.1 to 2.7 seconds (MPIA = 0.005 second). Statistics are computed over ten trials: (a) average throughput; (b) 90th percentile of the packet delays empirical distribution.

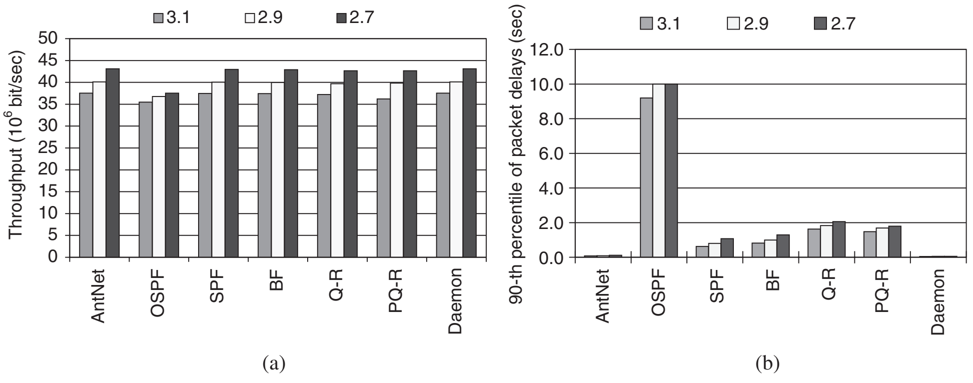
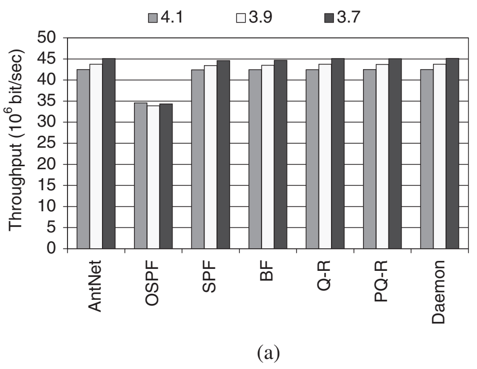


Figure 6.11
NTTnet: comparison of algorithms for increasing load for RP traffic. The load is increased reducing the MSIA value from 3.1 to 2.7 seconds (MPIA = 0.005 second). Statistics are computed over ten trials: (a) average throughput; (b) 90th percentile of the packet delays empirical distribution.



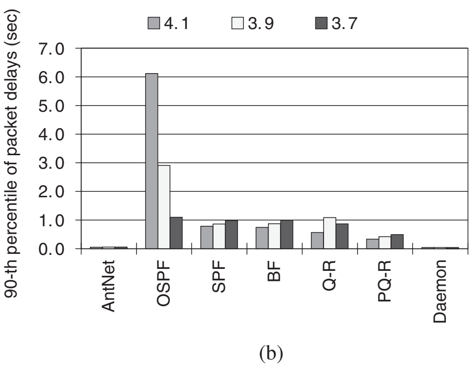


Figure 6.12
NTTnet: comparison of algorithms for increasing load for UP-HS traffic. The load is increased reducing the MSIA value from 4.1 to 3.7 seconds (MPIA = 0.3 second, HS = 4, MPIA-HS = 0.05 second). Statistics are computed over ten trials: (a) average throughput; (b) 90th percentile of the packet delays empirical distribution.

at least one order of magnitude in favor of AntNet. AntNet keeps delays at low values, very close to those obtained by Daemon, whereas SPF, BF, Q-R, and PQ-R perform poorly and OSPF completely collapses.

Note that in the UP-HS case, OSPF, which is the worst algorithm in this case, shows an interesting behavior. The increase in the generated data throughput determines a decrease or a very slow increase in the delivered throughput while delays decrease (see figure 6.12). In this case the load was too high for the algorithm and the balance between the two, conflicting, objectives, throughput and packet delays, showed an inverse dynamics: having a lot of packet losses made it possible for the surviving packets to obtain lower trip delays.

The TMPHS-UP experiment (figure 6.13), concerning sudden load variation, confirms the previous results. OSPF is not able to follow properly the variation both for throughput and delays. All the other algorithms were able to follow the sudden increase in the input throughput, but only AntNet (and Daemon) show a very regular behavior. Differences in packet delays are striking. AntNet performance is very close to that obtained by Daemon (the curves are practically superimposed at the scale used in the figure). Among the other algorithms, SPF and BF are the best, although their response is rather irregular and, in any case, much worse than AntNet's. OSPF and Q-R are out-of-scale and show a very delayed recovering curve. PQ-R, after a huge jump, which takes the graph out-of-scale in the first 40 seconds after hot spots are switched on, shows a trend approaching that of BF and SPF.

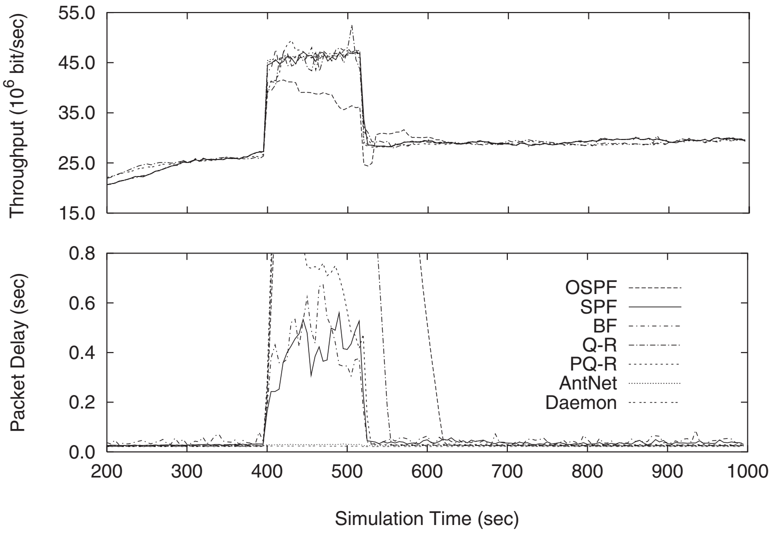


Figure 6.13
NTTnet: comparison of algorithms for transient saturation conditions with TMPHS-UP traffic (MSIA = 4.0 second, MPIA = 0.3 second, HS = 4, MPIA-HS = 0.05). Statistics are computed over ten trials: (up) average throughput; (down) packet delays averaged over 5-second moving windows. Reprinted by permission from Di Caro & Dorigo (1998c), © Morgan Kaufmann Publishers.

6.4.3 Routing Overhead

Table 6.2 reports results concerning the overhead generated by the routing packets. For each algorithm, the network load generated by the routing packets is reported as the ratio between the bandwidth occupied by the routing packets and the total available network bandwidth. Each row in the table refers to one of the experiments discussed in the two previous subsections. Routing overhead is computed for the experiment with the heaviest load in the increasing load series.

All data are scaled by a factor of 10^{-3}. The data in the table show that the routing overhead is negligible for all the algorithms with respect to the available bandwidth. Among the adaptive algorithms, BF shows the lowest overhead, closely followed by SPF. AntNet generates a slightly bigger consumption of network resources, but this is widely compensated by the higher performance it provides. Q-R and PQ-R produce an overhead a bit higher than that of AntNet. The routing load caused by the different algorithms is a function of many factors, specific to each algorithm. Q-R and PQ-R are data-driven algorithms: if the number of data packets or the length of the followed paths (because of topology or bad routing) grows, so will the number of

Table 6.2
Routing overhead: ratio between the bandwidth occupied by the routing packets and the total available network bandwidth

	AntNet	OSPF	SPF	BF	Q-R	PQ-R
NSFnet—UP	2.39	0.15	0.86	1.17	6.96	9.93
NSFnet—RP	2.60	0.15	1.07	1.17	5.26	7.74
NSFnet—UP-HS	1.63	0.15	1.14	1.17	7.66	8.46
NTTnet—UP	2.85	0.14	3.68	1.39	3.72	6.77
NTTnet—RP	4.41	0.14	3.02	1.18	3.36	6.37
NTTnet—UP-HS	3.81	0.14	4.56	1.39	3.09	4.81

All data are scaled by a factor of 10^{-3}. Adapted from Di Caro & Dorigo (1998c).

generated routing packets. BF, SPF, and OSPF have a more predictable behavior: the generated overhead is mainly a function of the topologic properties of the network and of the generation rate of the routing information packets. AntNet produces a routing overhead function of the ants' generation rate and of the length of the paths they travel.

The ant traffic can be roughly characterized as a collection of additional traffic sources, one for each network node, producing very small packets (and related acknowledgment packets) at a constant bit rate with destinations matching the input data traffic. On average, ants will travel over rather "short" paths and their size will grow by only 8 bytes at each hop. Therefore, each "ant routing traffic source" represents a very light additional traffic source with respect to network resources when the ant launching rate is not excessively high. In figure 6.14, the sensitivity of AntNet with respect to the ant launching rate is reported for a sample case of a UP data traffic model on NSFnet (previously studied in figure 6.6). The interval Δt between two consecutive ant generations is progressively decreased (Δt is the same for all nodes). Δt values are sampled at constant intervals over a logarithmic scale ranging from about 0.006 to 25 seconds. The lower, dashed, curve interpolates the generated routing overhead expressed, as before, as the fraction of the available network bandwidth used by routing packets. The upper, solid, curve plots the data for the obtained power normalized to its highest value observed during the trials, where the power is defined as the ratio between the delivered throughput and the 90th percentile of the packet delay distribution. The value used for delivered throughput is the throughput value at time 1000 averaged over ten trials, while for packet delay the 90th percentile of the empirical distribution was used.

In figure 6.14, it can be seen how an excessively small Δt causes an excessive growth of the routing overhead, with consequent reduction of the algorithm power. Similarly, when Δt is too big, the power slowly diminishes and tends toward a

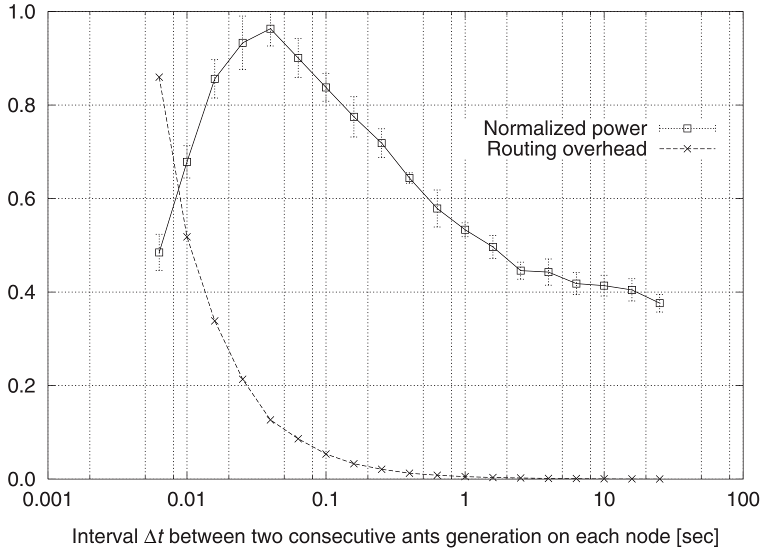


Figure 6.14
AntNet normalized power versus normalized routing overhead as a function of the interval Δt between two consecutive ants generation. Power is defined as the ratio between delivered throughput and the 90th percentile of the distribution of packet delays, and it is normalized to its highest value observed during the trials. Routing overhead is normalized by taking the ratio between the bandwidth used by the artificial ants and the available bandwidth over the whole network. Adapted from Di Caro & Dorigo (1998c).

plateau because the number of ants is not enough to generate and maintain up-to-date statistics of the network status. In the middle of these two extreme regions a wide range of Δt intervals gives rise to similar, very good power values, while, at the same time, the routing overhead quickly falls toward negligible values. It should be noted that the value for Δt used in the experiments, $\Delta t = 0.3$, is not optimal. The reason for using a suboptimal parameter is that the analysis, whose results are shown in figure 6.14, was performed only after all the experiments were run, and the already good results obtained with the suboptimal parameter value did not motivate the authors to re-run the experiments.

6.5 AntNet and Stigmergy

In AntNet, the continual online adaptation of pheromone matrices (and therefore of the corresponding routing tables) is the emerging result of a collective learning process. In fact, each forward-backward ant pair is complex enough to find a good route and to adapt the pheromone matrices for a single-source destination path, but it cannot solve the global routing optimization problem. It is the interaction between the ants that determines the emergence of a global effective behavior from the point of view of network performance. Ants cooperate in their problem-solving activity by communicating in an indirect and noncoordinated asynchronous way. Each ant acts independently. Good routes are discovered by applying a policy that is a function of the information accessed through the network nodes visited, and the information collected about the route is eventually released on the same nodes. Therefore, communication among artificial ants is mediated in an explicit and implicit way by the "environment," that is, by the node's data structures and by the traffic patterns recursively generated by the data packets' utilization of the routing tables. In other words, ants exploit stigmergic communication (see chapter 1, section 1.4, for a definition of stigmergy). The stigmergic communication paradigm matches well the intrinsically distributed nature of the routing problem.

Cooperation among artificial ants goes on at two levels: (1) by modifications of the pheromone matrices, and (2) by modifications of local models that determine the way the ants' performance is evaluated. The way pheromone matrices are modified depends, among others, on the value of the reinforcement r. As we have seen in section 6.2.3, in AntNet this value is set to be a function of the ant's trip time and of the node-local statistical models [according to equations (6.7), (6.8), and (6.9)]. It is interesting, however, to note that reasonably good results are obtained when setting the value r to a constant. Results of experiments run with this strategy are presented in figure 6.15. These results suggest that the "implicit" component of the algorithm, based on the ant arrival rate (differential path length effect), plays a very important role. Of course, to compete with state-of-the-art algorithms, the available information about path costs has to be used.

As shown in the previous section, the results obtained with the above stigmergic model of computation are excellent. In terms of throughput and average delay, AntNet performs better than both classic and recently proposed routing algorithms on a wide range of experimental conditions (see Di Caro & Dorigo, 1998a,b,c,e,f, for further experimental results).

Finally, it is interesting to remark that the used stigmergy paradigm makes AntNet's artificial ants very flexible from a software engineering point of view. In this perspective, once the interface with the node's data structure is defined, the internal policy of the ants can be transparently updated. Also, the ants could be upgraded to become richer mobile agents that carry out multiple concurrent tasks as, for example, collecting information for distributed network management or for Web data-mining tasks (see Di Caro, 2003, for the first results in this direction).

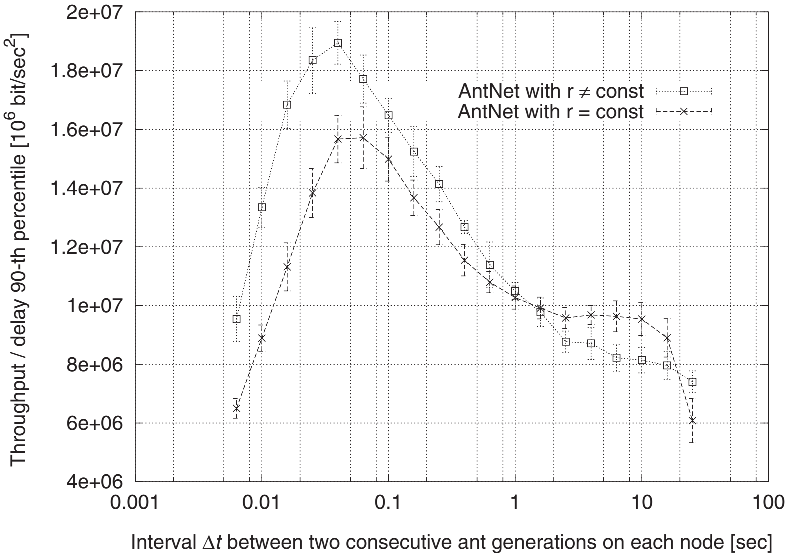


Figure 6.15
AntNet power for constant ($r = const$) and quality-based ($r \neq const$) pheromone updates as a function of the interval Δt between two consecutive ants generation (ants' launching rate). Power is defined as the ratio between delivered throughput and the 90th percentile of the distribution of packet delays.

6.6 AntNet, Monte Carlo Simulation, and Reinforcement Learning

The structure of AntNet allows one to draw some parallels with both parallel Monte Carlo simulation and with some well-known reinforcement learning (RL) algorithms. This is what is discussed in the rest of this section.

6.6.1 AntNet as an Online Monte Carlo System with Biased Exploration

The AntNet routing system can be seen as a collection of mobile agents collecting data about network status by concurrently performing online Monte Carlo simulations (Rubinstein, 1981; Streltsov & Vakili, 1996). In Monte Carlo methods, repeated experiments with stochastic transition components are run to collect data about the statistics of interest. Similarly, in AntNet, ants explore the network by performing random experiments (i.e., building paths from source to destination nodes using a stochastic policy dependent on the past and current network states), and collect online information on network status. A built-in variance reduction effect is determined by the way ants' destinations are assigned, biased by the most frequently observed data destinations, and by the way the ants' policy makes use of current and past traffic information. In this way, the explored paths match the most interesting paths from a data traffic point of view, which results in a very efficient variance reduction effect in the stochastic sampling of the paths. Unlike the usual offline Monte Carlo systems, in AntNet the state space sampling is performed online, that is, the sampling of the statistics and the controlling of the nonstationary traffic process are performed concurrently.

This way of exploring the network concurrently with data traffic is very different from what happens in the other algorithms where there is either no exploration at all (OSPF, SPF, and BF), or exploration is both tightly coupled to data traffic and of a local nature (Q-R and PQ-R). As shown in section 6.4.3, the extra traffic associated with the exploration is negligible for a wide range of values which allow very good performance.

6.6.2 AntNet and Reinforcement Learning

The characteristics of the routing problem allow one to interpret it as a distributed, stochastic time-varying RL problem (Sutton & Barto, 1998). This fact, as well as the structure of AntNet, makes it natural to draw some parallels between AntNet and classic RL approaches.

A first way to relate the structure of AntNet to that of an RL algorithm is connected to the way the outcomes of the experiments, the trip times $T_{k \to d}$, are processed. The transformation from the raw values $T_{k \to d}$ to the more refined reinforcements r are reminiscent of what happens in *actor-critic* systems (Barto, Sutton, & Anderson, 1983): the raw feedback signal from the environment is processed by a *critic* module, which is learning a model (the node's component $\mathcal{M}$) of the underlying process, and then is fed to the *actor* module that implements the policy (the pheromone matrix $\mathcal{T}$) and updates it according to the *critic* signal which consists of an evaluation of the policy followed by the ants. In our case, the *critic* is both adaptive, to take into account the variability of the traffic process, and rather simple, to meet computational requirements.

Another way of seeing AntNet as a classic RL system is related to its interpretation as a parallel replicated Monte Carlo system, as discussed in the previous subsection. In fact, as was shown first by Singh & Sutton (1996), a first-visit Monte Carlo simulation system (only the first visit to a state is used to estimate its value during a trial) is equivalent to a batch temporal difference (TD) method (Sutton, 1988) with replacing traces and decay parameter $\lambda = 1$. Although AntNet is a first-visit Monte Carlo simulation system, there are some important differences with the type of Monte Carlo used by Singh and Sutton (and in other RL works), mainly due to differences in the considered class of problems. In AntNet, outcomes of experiments are used both to update local models able to capture the variability of the whole network status (only partially observable) and to generate a sequence of stochastic policies. On the contrary, in the Monte Carlo system considered by Singh and Sutton, outcomes of the experiments are used to compute maximum-likelihood estimates of the expected mean and variance of the states' returns (i.e., the total reward following a visit of a state) of a Markov chain.

In spite of these differences, the weak parallel with TD(λ) methods is rather interesting, and allows highlighting an important difference between AntNet and its competitors (and general TD methods): in AntNet there is no backchaining of the information from one state (i.e., a triple [current node, destination node, next hop node]) to its predecessors. Each state is rewarded only on the basis of the ant's trip time information strictly relevant to it. This approach is completely different from that followed by Q-R, PQ-R, and BF, which are TD methods, and, from a different perspective, by SPF. In fact, these algorithms build the distance estimates at each node by using the predictions made at other nodes. In particular, Q-R and PQ-R, which propagate the estimation information only one step back, are precisely distributed versions of the TD(0) class of algorithms. They could be transformed into generic TD(λ), $0 < \lambda \leq 1$, by transmitting backward to all the previously visited nodes the information collected by the routing packet generated after each data hop. Of course, this would greatly increase the routing traffic generated, because it has to be done after each hop of each data packet, making the approach at least very costly, if feasible at all.

In general, using temporal difference methods in the context of routing presents an important problem: the key condition of the method, the *self-consistency between the estimates of successive states*, may not be strictly satisfied in the general case. Here, by self-consistency between the estimates of successive states, we mean, for example, that the prediction made at node k about the time to go to the destination node d should be additively related to the prediction for the same destination from each one of k's neighbors, each neighbor being one of the ways to go to d. The lack of self-consistency in routing applications is due to the fact that (1) the dynamics at each node is related in a highly nonlinear way to the dynamics of all its neighbors, (2) the traffic process evolves concurrently over all the nodes, and (3) there is a recursive interaction between the traffic patterns and the control actions (i.e., the modifications of the pheromone matrices). This aspect can explain in part the poor performance of the pure TD(0) algorithms Q-R and PQ-R.

6.7 Bibliographical Remarks

This chapter is strongly based on the work presented in Di Caro & Dorigo (1998c). Additional results obtained with AntNet on different network topologies can be found in a number of publications by Di Caro & Dorigo (1998a,b,e,f). A recent extension of AntNet in the direction of allowing network resources reservation is AntNet++. AntNet++ is a multiagent architecture for distributed learning and control in networks providing at the same time several types of services (e.g., best-effort and resource reservation at the same time) and is described in Dr. Di Caro's doctoral thesis (Di Caro, in preparation).

Schoonderwoerd and colleagues (1996) were the first to consider routing as a possible application domain for ACO algorithms. Their ant-based control (ABC) approach, which was applied to routing in telephone networks, differs from AntNet in many respects. The main differences are a direct consequence of the different network model they considered, which has the following characteristics: (1) connection links potentially carry an infinite number of full-duplex, fixed bandwidth channels, and (2) transmission nodes are crossbar switches with limited connectivity (i.e., there is no necessity for queue management in the nodes). In such a model, bottlenecks are put on the nodes, and the congestion degree of a network can be expressed in terms of connections still available at each switch. As a result, the network is cost-symmetric: the congestion status over available paths is completely bidirectional. The path $(n_0, n_1, n_2, \ldots, n_k)$ connecting nodes n_0 and n_k will exhibit the same level of congestion in both directions because the congestion depends only on the state of the nodes in the path. Moreover, dealing with telephone networks, each call occupies exactly one physical channel across the path. Therefore, "calls" are not multiplexed over the links, but they can be accepted or refused, depending on the possibility of reserving a physical circuit connecting the caller and the receiver. All these modeling assumptions make the problem of Schoonderwoerd et al. very different from the cost-asymmetric routing problem for data networks discussed in this chapter. This difference is reflected in many algorithmic differences between ABC and AntNet, the most important of which is that in ABC ants update pheromone trails after each step, without waiting for the completion of an ant's trip, as done in AntNet. This choice, which makes ABC behavior closer to real ants' behavior and which is reminiscent of the pheromone trail updating strategy implemented in the ant-density variant of AS (see chapter 3, section 3.3.1), was made possible by the cost-symmetry assumption made by the authors. Other differences are that ABC does not use local models to score the ants' trip times, nor local heuristic information and ant-private memory to improve the ants' decision policies. Also, it does not recover from cycles and does not use the information contained in all the ant subpaths.

Subramanian and colleagues (1997) have proposed an ant-based algorithm for packet-switched nets. Their algorithm is a straightforward extension of ABC, obtained by adding so-called *uniform ants*, an additional exploration mechanism that should avoid a rapid suboptimal convergence of the algorithm. A major limitation of these authors' work is that, although the algorithm they propose is based on the same cost-symmetry hypothesis as ABC, they apply it to packet-switched networks where this requirement is very often not met.

Heusse, Snyers, Guérin, & Kuntz (1998) have proposed the Cooperative Asymmetric Forward (CAF) model for routing in networks with asymmetric costs. CAF is an ant-based approach that is intended to build routing tables that permit use of a collection of paths between each pair of nodes. Unlike what happens in ABC and AntNet, in CAF routing tables are based on cost estimates to destination nodes. Depending on the objective pursued, CAF may use different metrics. Originally it was studied using the delay metric, but it was later more deeply studied using the load metric in the context of connection-oriented networks (Heusse & Kermarrec, 2000). CAF is strongly focused on convergence speed, and a complete presentation of this technique, which involves a large amount of cooperation between two types of agents, may be found in Heusse (2001).

Recently there has been a surge in interest concerning the use of ACO, and in particular of AntNet-like, algorithms for routing in mobile ad hoc networks (MANETs). Preliminary results, presented, for example, in Fujita, Saito, Matsui, & Matsuo (2002), Güneş, Sorges, & Bouazizi (2002), Güneş & Spaniol (2002), Baras & Mehta (2003), and Heissenbüttel & Braun (2003), are very promising and suggest that routing problems in these highly dynamic types of networks are a possible novel area for the successful application of ACO.

6.8 Things to Remember

- Network routing is a difficult problem because of its stochastic and time-varying nature.
- The distributed nature of network routing is well matched by the multiagent nature of ACO algorithms.
- AntNet is an ACO algorithm especially designed to solve routing problems in data networks. Its main differences with classic ACO algorithms applied to $\mathcal{NP}$-hard problems are (1) its use of the network graph as the construction graph; (2) its asyn-

chronous nature (ants do not move synchronously on the graph as in $\mathcal{NP}$-hard applications), which allows the exploitation of the differential path length effect observed in real ants; (3) the extra machinery required to evaluate meaningfully the quality of the paths produced by the ants; and (4) the fact that it is used to solve online problems.

▪ In the simulation conditions and on the problems described in section 6.3, AntNet reaches a performance that is comparable to, or better than that of state-of-the-art algorithms such as OSPF, SPF, adaptive Bellman-Ford, Q-Routing, and PQ-Routing:

• Under *low load conditions*, all the algorithms tested have similar performance. In this case, also considering the huge variability in the possible traffic patterns, it is very hard to assess whether an algorithm is significantly better than another or not.

• Under *high, near-saturation loads*, all the tested algorithms are able to deliver the input throughput in a quite similar way, that is, in most of cases all the generated traffic is routed without big losses. On the contrary, the study of packet delay distributions has shown remarkable differences among the algorithms, in favor of AntNet.

• Under *saturation*, packet losses or packet delays, or both, become too big, causing the network operations to slow down. Therefore, saturation has to be only a temporary situation. If it is not, structural changes to the network characteristics, like adding new and faster connection lines, rather than improvements of the routing algorithm, should be in order. For these reasons, the responsiveness of the algorithms to traffic loads causing only a temporary saturation was studied. Here also, AntNet had a better performance than the competing algorithms.

6.9 Computer Exercises

Exercise 6.1 Reimplement the AntNet algorithm using as network traffic simulator a public domain software such as, for example, OMNeT++ or NS2. OMNeT++ is available at whale.hit.bme.hu/omnetpp/; NS2 is available at www.isi.edu/nsnam/ns/.

Exercise 6.2 In AntNet, at each intermediate node visited while building a path to their destination node, forward ants wait in line in the data packet queues. Although this allows them to simulate exactly the behavior of data packets, it causes delays in the subsequent propagation of the collected information (to be done by the corresponding backward ants). A possibility of avoiding this inherent delay would be to let forward ants use the same high-priority queues used by backward ants and to let backward ants make use of the current status of the local-link queues, that is, the number of bits waiting to be sent, to estimate the time that would have been required for a forward ant to cross the link at the current moment using the data queues. In this way, forward ants are much quicker in building a path from source to destination, and at the same time the backward ants update the local models and the routing tables with more up-to-date information.

Implement a variant of AntNet in which artificial ants use the above-mentioned estimates to evaluate the quality of their paths and compare it to the standard AntNet. Do you expect an increase or a decrease in performance? Are the results you obtain a function of the degree of variability of the data traffic?
Hint: You can find a discussion of this extension of AntNet in Di Caro & Dorigo (1998f), where it is called AntNet-CO, and in Di Caro (2003), where it is called AntNet-FA.

Exercise 6.3 Investigate the behavior of AntNet when each artificial ant can have a different value for the parameter α [α, found in equation (6.3), weighs the importance of the heuristic values with respect to the pheromone values].

Exercise 6.4 Study the behavior of AntNet when changing the values of the parameters listed in box 6.1.

Exercise 6.5 In AntNet, whenever an ant uses a link, the associated pheromone is incremented. Try to implement a version of AntNet in which ants can also cause a decrease in pheromones (i.e., negative updates are possible). The idea is that if an ant generates a very bad path, it could be sensible to decrease the probability of choosing it by future ants. Compare this version of AntNet with the standard one.

Exercise 6.6 Test the behavior of AntNet in the absence of data traffic. Check that it converges to routing tables implementing shortest paths among all node pairs. Does convergence to shortest paths depend on the way the reinforcement r is computed?

7 Conclusions and Prospects for the Future

Go to the ant, thou sluggard; consider her ways, and be wise. Which having no guide, overseer, or ruler, Provideth her meat in the summer, and gathereth her food in the harvest.
—Proverbs 6: 6–8

At the time of completing this monograph on ant colony optimization (early summer 2003), it was 13 years since the first ideas that led to ACO were developed at the Politecnico di Milano in Milan, Italy, and just 4 years since ACO was formalized as a metaheuristic. In this short time span many things have happened and ACO is now a well-recognized member of the family of metaheuristic methods for discrete optimization problems.

In this final chapter we briefly summarize what we know about ACO and we give a short overview of the current main research trends. We conclude by putting ACO in the context of the wider research field of *ant algorithms*.

7.1 What Do We Know about ACO?

What we have learned in the first 13 years of life of ACO is a lot. Even more is what we still need to learn and discover. In this section we briefly summarize our current knowledge of the ACO metaheuristic, and in the next section we overview what are, in our opinion, the most promising current research trends.

7.1.1 Theoretical Developments

To repeat the epigraph at the beginning of chapter 4: *In theory, there is no difference between theory and practice. But in practice, there is a difference!* Apart from being amusing, this aphorism contains much wisdom. It is true, in fact, that the theory developed for ACO (the same is true for other metaheuristics, though) has little use in practical terms. Nevertheless, what we have learned about theory will hopefully be useful for better understanding the working of our algorithms and, maybe, for designing better-performing ones in the future.

Summarizing, what we know about the theory aspects of ACO is the following:

- *Convergence proofs.* We know that some of the best-performing ACO algorithms ($\mathcal{MM}$AS and ACS), both with and without local search, converge in value. As explained in chapter 4, convergence in value means that they will find, sooner or later, the optimal solution. This is a rather weak result, since it also applies to random search, and you can force the same property on any ACO algorithm by adding, for example, a procedure called every constant number of steps and that generates a random solution. The interesting point is, however, that the proof applies to two algorithms, $\mathcal{MM}$AS and ACS, that were not designed to converge, and that at the same time have good performance over many different combinatorial optimization problems.

We also know that it is possible to force an ACO algorithm to converge in solution (i.e., generate over and over the same, optimal solution). This result can be obtained by letting the pheromones evaporate very slowly, so that the optimal solution has probability 1 of being generated before it might become impossible to generate it. This result has only theoretical interest. In fact, in optimization we are interested in generating the optimal solution once, and the fact that the algorithm generates it over and over has no practical interest.

- *Model-based search framework.* An interesting question when a new algorithm or a new metaheuristic is proposed is its relation to other already existing algorithms or metaheuristics. By putting ACO in the framework of model-based search, a first, rather general answer to this question has been given: ACO algorithms have some general characteristics in common with such different algorithms as population based incremental learning (Baluja & Caruana, 1995), mutual-information–maximizing input clustering (MIMIC) (De Bonet et al., 1997), cross-entropy (Rubinstein, 1999), stochastic gradient descent (Robbins & Monroe, 1951), and estimation of distribution algorithms (Larrañaga & Lozano, 2001).

7.1.2 Experimental Results and Real-World Applications

As we have seen in chapter 5, ACO algorithms have been tested on a large number of academic problems. These include problems related to the traveling salesman, as well as assignment, scheduling, subset, and constraint satisfaction problems. For many of these, world-class performance has been achieved. For example, ACO algorithms are, at the time of writing, state-of-the-art (i.e., their performance is comparable to, or better than, that of the best existing methods other than ACO) for the sequential ordering problem (Gambardella & Dorigo, 2000), the vehicle routing problem with time window constraints (Gambardella et al., 1999), the quadratic assignment problem (Maniezzo, 1999; Stützle & Hoos, 2000), the group shop scheduling problem (Blum, 2003a), the arc-weighted l-cardinality tree problem (Blum & Blesa, 2003), and the shortest common supersequence problem (Michel & Middendorf, 1999). Additionally, very good performance has been obtained by AntNet (see chapter 6) on network routing problems (Di Caro & Dorigo, 1998c).

This success with academic problems has raised the attention of a number of companies that have started to use ACO algorithms for real-world applications. Among the first to exploit algorithms based on the ACO metaheuristic is EuroBios (www.eurobios.com). They have applied ACO to a number of different scheduling problems such as a continuous two-stage flow shop problem with finite reservoirs. The modeled problem included various real-world constraints such as setup times, capacity restrictions, resource compatibilities, and maintenance calendars. Another company that has played, and still plays, a very important role in promoting the real-world application of ACO is AntOptima (www.antoptima.com). AntOptima's researchers have developed a set of tools for the solution of vehicle routing problems whose optimization algorithms are based on ACO. Particularly successful products based on these tools are (1) DYVOIL, for the management and optimization of heating oil distribution with a nonhomogeneous fleet of trucks, used for the first time by Pina Petroli in Switzerland, and (2) ANTROUTE, for the routing of hundreds of vehicles of Migros, the leading Swiss supermarket chain. Still another vehicle routing application was developed by BiosGroup for the French company Air Liquide. Other interesting real-world applications are those of Gravel, Price, & Gagné (2002), who have applied ACO to an industrial scheduling problem in an aluminum casting center, and by Bautista & Pereira (2002), who successfully applied ACO to solve an assembly line balancing problem with multiobjective function and constraints between tasks for a bike assembly line.

7.2 Current Trends in ACO

Today, several hundred papers have been written on the applications of ACO. It is a true metaheuristic, with dozens of application areas. While both the performance of ACO algorithms and our theoretical understanding of their working have significantly increased, as shown in previous chapters, there are several areas in which until now only preliminary steps have been taken and where much more research will have to be done.

One of these research areas is the extension of ACO algorithms to more complex optimization problems that include (1) *dynamic problems*, in which the instance data, such as objective function values, decision parameters, or constraints, may change while solving the problem; (2) *stochastic problems*, in which one has only probabilistic information about objective function value(s), decision variable values, or constraint boundaries, due to uncertainty, noise, approximation, or other factors; and (3) *multiple objective problems*, in which a multiple objective function evaluates competing criteria of solution quality.

Active research directions in ACO include also the effective parallelization of ACO algorithms and, on a more theoretical level, the understanding and characterization of the behavior of ACO algorithms while solving a problem.

7.2.1 Dynamic Optimization Problems

A dynamic problem is a problem defined as a function of some quantities whose value is set by the dynamics of an underlying system. In other words, some of the characteristics of the problem change over time. A paradigmatic example is network routing, as discussed in chapter 6 (it should be noted that the network routing problems discussed in chapter 6 are both dynamic and stochastic: dynamic because traffic changes over time, and stochastic because the value assumed by traffic at different temporal instants is a stochastic variable).

Another dynamic problem that has been considered in the literature on ACO is the dynamic traveling salesman problem. In the dynamic TSP, cities may be deleted or added over time (see chapter 2, section 2.3.6). Guntsch & Middendorf (2001) and Guntsch, Middendorf, & Schmeck (2001) consider the case in which a certain percentage of the cities are deleted and replaced by new cities. The problem is then how to recompute quickly a good tour for the new TSP problem. Guntsch and colleagues propose three strategies. The first consists of a simple restart of the algorithm after all pheromone trails are reinitialized. The second and third strategies are based on the hypothesis that, at least for dynamic TSPs in which the percentage of cities replaced is not too high, it is useful to exploit the information contained in the pheromone trails. One of the two strategies reinitializes the pheromone trails exploiting heuristic information, while the other makes use of pheromone information. The experimental results, conducted on two instances from the TSPLIB (Reinelt, 1991), show that, if given enough computation time, the simple restart strategy performs better. Otherwise, if the time between two insertions/deletions is short, then the simple restart strategy has not enough time to find a new solution and the two restart strategies that partially reuse the pheromone information perform better.

Guntsch & Middendorf (2002b) have also proposed the use of *population-based* ACO for both dynamic TSPs and dynamic QAPs, where the dynamic QAP they consider is a QAP where in each fixed number of iterations some percentage of the locations is deleted and replaced by new locations. Population-based ACO differs from standard ACO as described in chapter 3 in that it maintains a population of *pop* solutions, used for updating the pheromone matrix. The algorithm works as follows: At the start of the algorithm, the population is empty. For the first *pop* iterations the iteration-best solution is added to the growing population without any constraints and no solution leaves the population. Whenever a solution enters the population, the pheromone matrix (which is initialized uniformly with a value τ_0) is updated by adding a constant quantity of pheromone $\Delta\tau$ to each element of the pheromone matrix which was used to build the solution. Beginning with iteration

$pop + 1$, the iteration-best solution becomes a candidate for insertion in the population. Guntsch and Middendorf propose a few mechanisms to decide whether to insert the iteration-best solution or not. These are based on simple heuristics such as removing the oldest solution in the population, or removing the worst one, or removing solutions with a probability inversely proportional to their quality, and other combinations thereof. When a solution is removed from the population, the pheromone matrix is updated by removing a constant quantity of pheromone $\Delta\tau$ from each element of the pheromone matrix which was originally used to build the removed solution.

The interesting point of using population-based ACO for dynamic problems is that, because all the information necessary to generate the pheromone matrix is maintained in the population, in case the problem instance dynamically changes, it is easy to apply a repair operator to the solutions in the population and then to regenerate the pheromone matrix using the repaired solutions. In Guntsch & Middendorf (2002a) some preliminary tests of this idea are run on two problem instances, one TSP and one QAP. Comparisons with a restart algorithm, that is, an algorithm that does not make any repair, but simply restarts after each modification in the problem instance, showed that the population-based ACO approach is competitive either when the changes to the problem are large (i.e., many cities/locations are substituted so that the new optimal solution is very different from the old one), or when the time interval between two changes is short, so that the restart algorithm has not enough time to find new good solutions.

Last, Eyckelhof & Snoek (2002) have considered another type of dynamic TSP in which the number of cities remains constant and what changes is the distance between some pairs of cities (this is intended to represent sudden changes in traffic between selected locations). Their preliminary experimental results show that AS, as well as the few extensions they propose, work reasonably well on some simple test problems.

7.2.2 Stochastic Optimization Problems

By stochastic optimization we mean here those optimization problems for which some of the variables used to define them have a stochastic nature. This could be the problem components, as defined in chapter 2, section 2.2.1, which can have some probability of being part of the problem or not, or the values taken by some of the variables describing the problem, or the value returned by the objective function.

To the best of our knowledge, the only stochastic optimization problems to which ACO has been applied are (1) network routing, which was already discussed at length in chapter 6 and which is also a dynamic optimization problem, and (2) the probabilistic TSP (PTSP).

The first application of ACO to the PTSP was done by Bianchi, Gambardella, & Dorigo (2002a,b). In the PTSP, a TSP problem in which each city has a given probability of requiring a visit, the goal is to find an a priori tour of minimal expected length over all the cities, with the strategy of visiting a random subset of cities in the same order as they appear in the a priori tour. The ACO algorithm chosen by Bianchi et al. to solve the PTSP was ACS. In fact, they implemented two versions of ACS: the standard one (see chapter 3, section 3.4.1) and pACS, which differs from ACS only in the way the objective function is computed. In ACS, it is computed in the standard way for the case in which each city has probability 1 to require a visit, whereas in pACS the probabilities with which cities require a visit are taken into account. In practice, pACS uses the exact objective function, which can be computed in $\mathcal{O}(n^2)$ (Jaillet, 1985, 1988), while ACS uses an approximation of the objective function, which can be computed in $\mathcal{O}(n)$.

pACS was first experimentally shown to outperform some problem-specific heuristics, and it was then compared with ACS. The experimental results, which were run on homogeneous instances of the PTSP (i.e., all cities have the same probability of requiring a visit), show that pACS is the best among the two algorithms except for probabilities close to 1, in which case ACS is more efficient than pACS. This is due to the fact that the computation of the exact objective function is CPU–time-consuming, and this overhead is not justified in those cases in which the approximate objective function, which can be computed much faster, is close enough to the exact one.

More recently, Branke & Guntsch (2003) have considered two ways of improving the performance of ACO for the PTSP: they experimentally show that a coarse and fast-to-compute approximation of the exact objective function and the use of problem-specific heuristics to guide the ants during tour construction improve the algorithm performance.

7.2.3 Multiobjective Optimization Problems

Many problems from real-world applications require the evaluation of multiple, often conflicting, objectives. In such problems, which are called multiobjective optimization problems (MOOPs), the goal becomes to find a solution that gives the best compromise between the various objectives.

The selection of a compromise solution has to take into account the preferences of the decision maker. There are different ways to determine such compromise solutions. Under some mild assumptions on the preferences of the decision maker

(Steuer, 1986), compromise solutions belong to the set of efficient (or Pareto-optimal) solutions. A solution is called efficient if it is not dominated by any other solution, and the *Pareto-optimal set* is the set that contains all the efficient solutions. Hence, one possibility to solve MOOPs is to find the Pareto-optimal set, or at least a good approximation of it. The solutions in the set can then be given to the decision maker, who will choose among them according to personal criteria.

Differently, if the decision maker can give weights or priorities to the objectives before solving the problem, then the MOOP can be transformed into a single objective problem. In fact, in the first case the different objectives can be combined in a single objective given by their weighted sum, while in the second case the different solutions can be ordered according to priorities and compared lexicographically.

The first applications of ACO to the solution of multiobjective optimization problems are based on prioritized objectives. One such approach is the two-colony approach of Gambardella et al. (1999a) to the vehicle routing problem with time window constraints, which was presented in chapter 5, section 5.1.2.

A multicolony approach was also proposed by Mariano & Morales (1999) for the design of water irrigation networks. Their approach differs from that of Gambardella et al. in that (1) the first colony constructs only a partial solution to the problem that is completed to a full solution by the second colony; and (2) the solutions used to update the pheromone trails are all the nondominated solutions found after the second colony has completed the solution construction.

Other applications of ACO to MOOPs using prioritized objectives are those of T'kindt, Monmarché, Tercinet, & Laügt (2002) and Gravel et al. (2002). T'kindt et al. applied $\mathcal{MM}$AS to a biobjective two-machine permutation flow shop problem. Computational results showed that $\mathcal{MM}$AS yields excellent performance from a solution quality point of view. Gravel et al. applied an ACO algorithm to a four-objectives problem arising in a real-world scheduling problem for a aluminum casting center. They used the objectives to construct the heuristic information to be used by the ACO algorithm. For the pheromone update, only the most important objective was taken into account.

Doerner, Hartl, & Reimann (2001, 2003) used two cooperative ant colonies for solving biobjective transportation problems. They combined the two objective functions into a single one using a weighted sum approach. However, they used two colonies, which exploit different heuristic information for the solution construction; each of the heuristic information used takes into account one of the objectives. From an algorithmic point of view, the two approaches presented in Doerner, Hartl & Reimann (2001, 2003) differ mainly in the way information is exchanged between the colonies and in the way these colonies interact. Computational results suggest that the multiple-colony approach leads to improved performance when compared to the use of a single colony with single heuristic information.

Few ACO approaches exist that try to approximate the set of Pareto-optimal solutions. Doerner, Gutjahr, Hartl, Strauss, & Stummer (2003) apply an extension of ACS to a portfolio optimization problem. In their approach, for each of the objectives there is one pheromone matrix. An ant constructs a solution based on a weighted combination of the pheromone matrices; the weights used by each ant are chosen randomly when the ant is generated and kept fixed over its lifetime. After all ants have finished their solution construction, the pheromone matrices for each objective are updated by allowing the two ants with the best solutions for the corresponding objective to deposit pheromone. Experimental results obtained on instances with five and ten objectives showed that the proposed ACS algorithm performed better than the Pareto-simulated annealing proposed in Czyzak & Jaszkiewicz (1998) and the nondominated sorting genetic algorithm (Deb, 2001).

Iredi, Merkle, & Middendorf (2001) applied ACO algorithms to a biobjective single-machine total tardiness scheduling problem with changeover costs c_{ij} when switching from a job i to a job j. They used a multicolony ACO algorithm, in which each of the multiple colonies specializes in a different region of the Pareto front. Each colony uses two pheromone matrices, one corresponding to the total tardiness criterion and one to the changeover costs. For the solution construction, an ant uses a weighted combination of the pheromones and of the heuristic information with respect to the two criteria. After all ants of all colonies have completed their solution, the set of all nondominated solutions from all colonies is determined; only ants in this set are allowed to deposit pheromone. Experimental tests were done considering various possibilities for defining the region of the Pareto front to which the colonies specialize and the strategies for the pheromone update. This work was extended by Guntsch & Middendorf (2002b), who adapted population-based ACO to multi-objective problems and applied it to the same problem treated in the Iredi et al. paper as well as to a variant of this problem with four objectives (Guntsch & Middendorf, 2003).

7.2.4 Parallelization

Even when using metaheuristics, the solution of real-world optimization problems may require long computation times. Parallel implementations of ACO algorithms, for running on distributed (parallel) computing hardware, are therefore desirable.

ACO is inherently a distributed methodology which makes use of many individual and local procedures, so it is particularly suited to parallelization. Although a number of parallel versions of ACO have been implemented and tested in limited settings (see chapter 3, section 3.5), it is still an open question as to how to implement efficient parallel versions of ACO, and what type of performance improvement can be obtained over sequential versions.

An interesting research direction would also be to develop and test truly distributed ACO algorithms running on parallel hardware. In particular, ACO software running on Beowulf-style clusters of PCs (Sterling, Salmon, Becker, & Savarese, 1999) and GRID computing systems would be very useful to allow experimentation with real-world problems presenting multiobjective, dynamic, and stochastic characteristics, as discussed earlier.

Finally, a recent work by Merkle & Middendorf (2002b) has opened the way to implementations of ACO algorithms on *run-time reconfigurable processor arrays*.

7.2.5 Understanding ACO's Behavior

ACO algorithms are complex systems whose behavior is determined by the interaction of many components such as parameters, macroscopic algorithm components (e.g., the form of the probabilistic rule used by ants to build solutions, or the type of pheromone update rule used), and problem characteristics. Because of this, it is very difficult to predict their performance when they are applied to the solution of a novel problem.

Recently, researchers have started to try to understand ACO algorithm behavior by two typical approaches of science: (1) the study of the complex system under consideration in controlled and simplified experimental conditions, and (2) the study of the conditions under which the performance of the studied system degrades. Contributions along these two lines of research are briefly discussed in the following.

Study of ACO in Controlled and Simplified Experimental Conditions
The analysis of ACO algorithm behavior on simple problems is interesting because the behavior of the algorithm is not obscured by factors due to the complexity of the problem itself. A first such analysis was presented in chapter 1 (see also Dorigo & Stützle, 2001), where Simple-ACO was applied to the problem of finding the shortest path in a graph. The experimental results show that many algorithm components, which are essential to more advanced ACO models applied to challenging tasks, are also important for efficiently finding shortest paths.

In a similar vein, Merkle & Middendorf (2003b) apply ACO to the linear assignment problem, a permutation problem that is solvable in polynomial time (Papadimitriou & Steiglitz, 1982). By varying the cost matrix, they are able to generate classes of instances that differ in the number and structure of the optimal solutions. They tested three different ways of using the pheromones for the construction of a permutation and explored three different ways of filling the permutation (forward construction, backward construction, and assigning the positions in a random order). The experiments enabled the identification of situations in which particular construction rules fail to achieve good behavior and the explaination of why this can be the case.

Merkle & Middendorf (2002c) also proposed a deterministic model of ACO algorithms based on the ants' expected behavior and used it to model the dynamics of an ACO algorithm that uses iteration-best update when applied to a special type of permutation problem that consists of several, independent subproblems. They studied the behavior of the ACO model analytically and performed a fixed point analysis of the pheromone matrices, showing that the position of the fixed points in the state space of the system has a strong influence on the algorithm's optimization behavior.

Finally, Blum & Dorigo (2003, 2004) experimentally and theoretically studied the behaviour of AS applied to unconstrained binary problems, that is, binary problems for which the values of different decision variables are independent of each other. They were able to prove that, in this setting, the expected quality of the solutions generated by AS increases monotonically over time. Although their result cannot be transferred to the application of AS to constrained problems, in Blum & Dorigo (2003) they give empirical evidence that it holds for one of the most studied constrained problems: the TSP.

Study of the Conditions under which ACO Algorithm Performance Degrades

The study of problems in which a stochastic algorithm's performance degrades is an important research direction to understand an algorithm's behavior. Work in this direction has recently been done by Blum, Sampels, & Zlochin (2002), who have shown analytically that the expected quality of the solutions found by AS on particular instances of the arc-weighted l-cardinality tree problem (see chapter 5, section 5.4.3) may decrease over time. In their example this was the case because for the particular instance chosen there are two equally good competing solutions, and a third, bad solution is taking profit from this. However, in this particular case, the use of different update rules like iteration-best update would not lead to such behavior. This type of analysis is extended by Blum & Sampels (2002b) to ACO algorithms for the group shop scheduling problem (see also chapter 5, section 5.3.2). They show experimentally that for particular choices in the definition of the pheromone trails, the average quality of the solutions returned by ACO algorithms may decrease for a number of iterations, even if iteration-best update is used. A more detailed analysis showed that this effect becomes stronger when the problem becomes more constrained. Hence, the problem constraints, together with the chosen meaning given to the pheromone trails, determine how strong this detrimental effect, which Blum and Sampels call model bias, is.

7.3 Ant Algorithms

The ideas presented in this book are part of a growing discipline known collectively as *ant algorithms*. Ant algorithms have been defined in Dorigo et al. (2000a) and Dorigo (2001) as multiagent systems inspired by the observation of real ant colony behavior exploiting *stigmergy*.

Stigmergy, defined in chapter 1, section 1.4, plays an important role in ant algorithms research because the implementation of ant algorithms is made possible by the use of so-called *stigmergic variables*, that is, variables that contain the information used by the artificial ants to communicate indirectly. In some cases, as in the foraging behavior discussed in chapter 1 and at the base of ACO, the stigmergic variable is a specifically defined variable used by ants to adaptively change the way they build solutions to a considered problem.

But ant foraging is not the only social insect behavior that has inspired computer scientists and roboticists. Other examples, that we shall discuss only briefly here, are brood sorting and division of labor. In these cases, as discussed in the following, the stigmergic variable is one of the problem variables: a change in its value determines not only a change in the way a solution to the problem is built but also a direct change in the solution of the problem itself. A more comprehensive discussion of ant algorithms and stigmergy can be found in Bonabeau, Dorigo, & Theraulaz (1999, 2000) and in Dorigo et al. (2000a). In the following subsections we briefly describe some of the current directions in ant algorithms research.

7.3.1 Other Models Inspired by Foraging and Path Marking

As we know by now, the foraging behavior of ant colonies is at the basis of the ACO metaheuristic. But foraging, and more generally path-marking, behaviors have also inspired other types of algorithms. For example, Wagner, Lindenbaum, & Bruckstein proposed two algorithms for exploring a graph called, respectively, *Edge Ant Walk* (Wagner, Lindenbaum, & Bruckstein, 1996) and *Vertex Ant Walk* (Wagner, Lindenbaum, & Bruckstein, 1998) in which one or more artificial ants walk along the arcs of the graph, lay a pheromone trail on the visited arcs (or nodes), and use the pheromone trails deposited by previous ants to direct their exploration. Although the general idea behind the algorithm is similar to the one that inspired ACO, the actual implementation is very different. In the work of Wagner et al., pheromone trail is the stigmergic variable and is used as a kind of distributed memory that directs the ants toward unexplored areas of the search space. In fact, their goal is to cover the graph, that is, to visit all the nodes, without knowing the graph topology. They were able to prove a number of theoretical results, for example, concerning the time complexity for covering a generic graph. Also, they recently extended their algorithms (Wagner, Lindenbaum, & Bruckstein, 2000) so that they can be applied to dynamically changing graphs. A possible and promising application of this work is Internet search, where the problem is to keep track of the hundreds of thousands of pages added every day (Lawrence & Giles, 1998) (as well as of those that disappear).

7.3.2 Models Inspired by Brood Sorting

Brood sorting is an activity which can be observed in many ant species (e.g., in *Leptothorax unifasciatus* [Franks & Sendova-Franks, 1992], in *Lasius niger* [Chrétien, 1996], and in *Pheidole pallidula* [Deneubourg, Goss, Franks, Sendova-Franks, Detrain, & Chrétien, 1991]), which compactly cluster their smaller eggs and microlarvae at the center of the nest brood area and the largest larvae at the periphery of the brood cluster. Deneubourg et al. (1991) proposed a model of this phenomenon in which an ant picks up and drops an item according to the number of similar surrounding items. For example, if an ant carries a small egg, it will, with high probability, drop the small egg in a region populated by small eggs. On the contrary, if an ant is unloaded and finds a large larva surrounded by small eggs, it will, with high probability, pick up the larva. In all other situations the probability with which an ant picks up or drops an item is set to a small value.

Lumer & Faieta (1994) and Kuntz, Snyers, & Layzell (1999) have applied this model to the following clustering problem. Given are a set of points in an n-dimensional space and a metric d which measures the distance between pairs of points. The points must be projected onto the plane in such a way that if any two projected points in the plane are neighbors, their corresponding points in the n-dimensional space are neighbors under the metric d. The initial projection is random and the artificial ants then perform random walks on the plane and pick up or drop projected data items using rules from the model of Deneubourg et al. (1991). The results obtained are promising: they are qualitatively equivalent to those obtained by classic techniques like spectral decomposition or stress minimization (Kuntz et al., 1999), but at a lower computational cost. Recently, Handl & Meyer (2002) extended Lumer and Faieta's algorithm and proposed an application to the classification of Web documents and to their visualization in the form of topic maps (Fabrikant, 2000).

The model of Deneubourg et al. (1991) has also inspired a number of researchers in collective robotics who have implemented robotic systems capable of building clusters of objects without the need for any centralized control (Beckers, Holland, & Deneubourg, 1994; Martinoli & Mondada, 1998). Holland & Melhuish (1999) have extended the model of Deneubourg et al. so that it can be used by a colony of robots to sort objects.

In all these applications the stigmergic variable is represented by the physical distribution of the items: different configurations of the items determine different actions by the artificial agents.

7.3.3 Models Inspired by Division of Labor

In ant colonies individual workers tend to specialize in certain tasks (Robinson, 1992). Nevertheless, ants can adapt their behavior to the circumstances: a soldier ant can become a forager, a nurse ant a guard, and so on. Such a combination of specialization and flexibility in task allocation is appealing for multiagent optimization and control, particularly in resource or task allocation problems that require continuous adaptation to changing conditions. Robinson (1992) developed a threshold model in which workers with low response thresholds respond to lower levels of stimuli than do workers with high response thresholds. In this model the stimuli play the role of stigmergic variables.

A response threshold model of division of labor in which task performance reduces the intensity of stimuli has been used to solve dynamic task-scheduling problems (Bonabeau et al., 1999; Bonabeau, Sobkowski, Theraulaz, & Deneubourg, 1997a). When workers with low thresholds perform their normal tasks, the task-associated stimuli never reach the thresholds of the high-threshold workers. But if, for any reason, the intensity of task-associated stimuli increases, high-threshold workers engage in task performance. Bonabeau et al. (1999) and Campos, Bonabeau, Theraulaz, & Deneubourg (2000) present an application of these ideas to the problem of choosing a paint booth for trucks coming out of an assembly line in a truck factory. In this system each paint booth is considered an insect-like agent that, although specialized in one color, can, if needed, change its color (though it is expensive). The ant algorithm minimizes the number of booth setups (i.e., paint changeovers). This and similar scheduling and task allocation problems were also recently investigated by Nouyan (2002) and Cicirello & Smith (2001, 2003).

The threshold model was also used by Krieger & Billeter (2000) and Krieger, Billeter, & Keller (2000) to organize a group of robots. They designed a group of Khepera robots (miniature mobile robots aimed at "desktop" experiments [Mondada, Franzi, & Ienne, 1993]) to collectively perform an object-retrieval task.

In one of the experiments they performed, the objects were spread in the environment and the robots' task was to take them back to their "nest" where they were dropped in a basket. The available "energy" of the group of robots decreased regularly with time, but was increased when pucks were dropped into the basket. More energy was consumed during retrieval trips than when robots were immobile in the nest. Each robot had a threshold for the retrieval task: when the energy of the colony went below the threshold of a robot, the robot left the nest to look for objects in the environment. Krieger and Billeter's experiment has shown the viability of the threshold-based ant algorithm in a rather simple environment. Further experimentation is necessary to test the methodology on more complex tasks.

7.3.4 Models Inspired by Cooperative Transport

The behavior of ant colonies has also inspired roboticists interested in the design of distributed control algorithms for groups of robots (Martinoli & Mondada, 1998). An example of a task that has been used as a benchmark for ant algorithms applied to distributed robotics problems is cooperative box pushing (Kube & Zhang, 1994). In several ant species, when it is impossible for a single ant to retrieve a large item, nest mates are recruited to help through direct contact or chemical marking (Franks, 1986; Moffett, 1988; Sudd, 1965), implementing in this way a form of cooperative transport. The ants move around the item they want to carry, changing position and alignment until they succeed in carrying it toward the nest. An ant algorithm which reproduces the behavior of real ants in a group of robots whose task is to push a box toward a goal has been implemented and tested by Kube & Zhang (1994). Another example of application of ant algorithms is the related problem of pulling an object. This has been achieved (Dorigo, Trianni, Şahin, Labella, Gross, Baldassare, Nolfi; Deneubourg, Mondada, Floreano, & Gambardella, 2003) within the Swarm-bots project (www.swarm-bots.org), a project dedicated to the study of ant algorithms for autonomous robotics applications.

Appendix: Sources of Information about the ACO Field

There are a number of sources for information about the ACO field. The most important ones are listed in the following.

▪ *Webpages*

• www.aco-metaheuristic.org: These are the official webpages dedicated to collecting information about ACO.

• www.metaheuristics.org: These are the webpages of the "Metaheuristics Network" project. This European Union–funded project is dedicated to the theoretical analysis and experimental comparison of metaheuristics.

▪ *Software*. Software, distributed under the GNU license, is available at: www.aco-metaheuristic.org/aco-code/

▪ *Popular press*. ACO is often covered by the popular press. Pointers to popularization articles can be found at: www.aco-metaheuristic.org/aco-in-the-press.html

▪ *Mailing list*. A moderated mailing list dedicated to the exchange of information related to ACO is accessible at: www.aco-metaheuristic.org/mailing-list.html

▪ *Conferences and journals*

• "ANTS 2004—Fourth International Workshop on Ant Colony Optimization and Swarm Intelligence." The ANTS biannual series of workshops (see iridia.ulb.ac.be/~ants), held for the first time in 1998, is the oldest conference in the ACO and swarm intelligence fields.

• "From Worker to Colony: International Workshop on the Mathematics and Algorithms of Social Insects." This workshop was held for the first time in Cambridge, UK, in 2001, and the second workshop took place at the Georgia Institute of Technology, Atlanta, in December 2003.

• Special sessions or special tracks on ACO are organized in many conferences. Examples are the IEEE Congress on Evolutionary Computation (CEC) and the Genetic and Evolutionary Computation (GECCO) series of conferences.

• Papers on ACO can regularly be found in many other conferences such as "Parallel Problem Solving from Nature" conferences, INFORMS meetings, ECCO conferences, the Metaheuristics International Conference, the European Workshop on Evolutionary Computation in Combinatorial Optimization, and many others, and in many journals, such as *Artificial Life*; *Evolutionary Computation*; *IEEE Transactions on Systems, Man, and Cybernetics*; *IEEE Transactions on Evolutionary Computation*; *INFORMS Journal on Computing*; *Journal of Operations Research Society*; *European Journal of Operational Research*; and so on.

References

Aardal, K. I., van Hoesel, S. P. M., Koster, A. M. C. A., Mannino, C., & Sassano, A. (2001). Models and solution techniques for the frequency assignment problem. Technical report 01-40, Konrad-Zuse-Zentrum für Informationstechnik, Berlin.

Aarts, E. H. L., Korst, J. H. M., & van Laarhoven, P. J. M. (1997). Simulated annealing. In E. H. L. Aarts & J. K. Lenstra (Eds.), *Local Search in Combinatorial Optimization* (pp. 91–120). Chichester, UK, John Wiley & Sons.

Aarts, E. H. L., & Lenstra, J. K. (Eds.). (1997). *Local Search in Combinatorial Optimization*. Chichester, UK, John Wiley & Sons.

Abdul-Razaq, T. S., Potts, C. N., & Wassenhove, L. N. V. (1990). A survey of algorithms for the single machine total weighted tardiness scheduling problem. *Discrete Applied Mathematics*, *26*(2), 235–253.

Alaettinoğlu, C., Shankar, A. U., Dussa-Zieger, K., & Matta, I. (1992). Design and implementation of MaRS: A routing testbed. Technical report UMIACS-TR-92-103, CS-TR-2964, Institute for Advanced Computer Studies and Department of Computer Science, University of Maryland, College Park.

Anstreicher, K. M., Brixius, N. W., Goux, J.-P., & Linderoth, J. (2002). Solving large quadratic assignment problems on computational grids. *Mathematical Programming*, *91*(3), 563–588.

Applegate, D., Bixby, R., Chvátal, V., & Cook, W. (1995). Finding cuts in the TSP. Technical report 95-05, DIMACS Center, Rutgers University, Piscataway, NJ.

Applegate, D., Bixby, R., Chvátal, V., & Cook, W. (1998). On the solution of traveling salesman problems. *Documenta Mathematica, Extra Volume ICM III*, 645–656.

Applegate, D., Bixby, R., Chvátal, V., & Cook, W. (1999). Finding tours in the TSP. Technical report 99885, Forschungsinstitut für Diskrete Mathematik, University of Bonn, Germany.

Applegate, D., Cook, W., & Rohe, A. (2003). Chained Lin-Kernighan for large traveling salesman problems. *INFORMS Journal on Computing*, *15*(1), 82–92.

Ausiello, G., Crescenzi, P., Gambosi, G., Kann, V., Marchetti-Spaccamela, A., & Protasi, M. (1999). *Complexity and Approximation—Combinatorial Optimization Problems and Their Approximability Properties*. Berlin, Springer-Verlag.

Bacchus, F., Chen, X., van Beek, P., & Walsh, T. (2002). Binary vs non-binary constraints. *Artificial Intelligence*, *140*(1–2), 1–37.

Baird, L. C., & Moore, A. W. (1999). Gradient descent for general reinforcement learning. In M. Kearns, S. Solla, & D. Cohn (Eds.), *Advances in Neural Information Processing Systems, 11* (pp. 968–974). Cambridge, MA, MIT Press.

Balas, E., & Vazacopoulos, A. (1998). Guided local search with shifting bottleneck for job shop scheduling. *Management Science*, *44*(2), 262–275.

Baluja, S., & Caruana, R. (1995). Removing the genetics from the standard genetic algorithm. In A. Prieditis & S. Russell (Eds.), *Proceedings of the Twelfth International Conference on Machine Learning (ML-95)* (pp. 38–46). Palo Alto, CA, Morgan Kaufmann.

Baras, J. S., & Mehta, H. (2003). A probabilistic emergent routing algorithm for mobile ad hoc networks. In *Proceedings of WiOpt03: Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks. Sophia-Antipolis, France, INRIA*.

Barto, A. G., Sutton, R. S., & Anderson, C. W. (1983). Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE Transactions on Systems, Man, and Cybernetics*, *13*, 834–846.

Battiti, R., & Protasi, M. (2001). Reactive local search for the maximum clique problem. *Algorithmica*, *29*(4), 610–637.

Battiti, R., & Tecchiolli, G. (1994). The reactive tabu search. *ORSA Journal on Computing*, *6*(2), 126–140.

Bauer, A., Bullnheimer, B., Hartl, R. F., & Strauss, C. (2000). Minimizing total tardiness on a single machine using ant colony optimization. *Central European Journal for Operations Research and Economics*, *8*(2), 125–141.

Baum, E. B. (1986). Towards practical "neural" computation for combinatorial optimization problems. In J. S. Denker (Ed.), *Neural Networks for Computing*, vol. 151 (pp. 53–58). New York, American Institute of Physics Conference Proceedings.

Bautista, J., & Pereira, J. (2002). Ant algorithms for assembly line balancing. In M. Dorigo, G. Di Caro, & M. Sampels (Eds.), *Proceedings of ANTS 2002—From Ant Colonies to Artificial Ants: Third International Workshop on Ant Algorithms*, vol. 2463 of *Lecture Notes in Computer Science* (pp. 65–75). Berlin, Springer-Verlag.

Baxter, J. (1981). Local optima avoidance in depot location. *Journal of the Operational Research Society*, *32*, 815–819.

Beckers, R., Deneubourg, J.-L., & Goss, S. (1993). Modulation of trail laying in the ant *Lasius niger* (hymenoptera: Formicidae) and its role in the collective selection of a food source. *Journal of Insect Behavior*, *6*(6), 751–759.

Beckers, R., Holland, O. E., & Deneubourg, J.-L. (1994). From local actions to global tasks: Stigmergy and collective robotics. In R. Brooks & P. Maes (Eds.), *Artificial Life IV* (pp. 181–189). Cambridge, MA, MIT Press.

Bellman, R. (1958). On a routing problem. *Quarterly of Applied Mathematics*, *16*(1), 87–90.

Bellman, R., Esogbue, A. O., & Nabeshima, I. (1982). *Mathematical Aspects of Scheduling and Applications*. New York, Pergamon Press.

Bentley, J. L. (1992). Fast algorithms for geometric traveling salesman problems. *ORSA Journal on Computing*, *4*(4), 387–411.

Berger, B., & Leight, T. (1998). Protein folding in the hydrophobic-hydrophilic (hp) model is NP-complete. *Journal of Computational Biology*, *5*(1), 27–40.

Bertsekas, D. (1995a). *Dynamic Programming and Optimal Control*. Belmont, MA, Athena Scientific.

Bertsekas, D. (1995b). *Nonlinear Programming*. Belmont, MA, Athena Scientific.

Bertsekas, D., & Gallager, R. (1992). *Data Networks*. Englewood Cliffs, NJ, Prentice Hall.

Bertsekas, D., & Tsitsiklis, J. (1996). *Neuro-Dynamic Programming*. Belmont, MA, Athena Scientific.

Bianchi, L., Gambardella, L. M., & Dorigo, M. (2002a). An ant colony optimization approach to the probabilistic traveling salesman problem. In J. J. Merelo, P. Adamidis, H.-G. Beyer, J.-L. Fernández-Villacanas, & H.-P. Schwefel (Eds.), *Proceedings of PPSN-VII, Seventh International Conference on Parallel Problem Solving from Nature*, vol. 2439 of *Lecture Notes in Computer Science* (pp. 883–892). Berlin, Springer-Verlag.

Bianchi, L., Gambardella, L. M., & Dorigo, M. (2002b). Solving the homogeneous probabilistic traveling salesman problem by the ACO metaheuristic. In M. Dorigo, G. Di Caro, & M. Sampels (Eds.), *Proceedings of ANTS 2002—From Ant Colonies to Artificial Ants: Third International Workshop on Ant Algorithms*, vol. 2463 of *Lecture Notes in Computer Science* (pp. 176–187). Berlin, Springer-Verlag.

Birattari, M., Di Caro, G., & Dorigo, M. (2002a). Toward the formal foundation of ant programming. In M. Dorigo, G. Di Caro, & M. Sampels (Eds.), *Proceedings of ANTS 2002—From Ant Colonies to Artificial Ants: Third International Workshop on Ant Algorithms*, vol. 2463 of *Lecture Notes in Computer Science* (pp. 188–201). Berlin, Springer-Verlag.

Birattari, M., Stützle, T., Paquete, L., & Varrentrapp, K. (2002b). A racing algorithm for configuring metaheuristics. In W. B. Langdon, E. Cantú-Paz, K. Mathias, R. Roy, D. Davis, R. Poli, K. Balakrishnan, V. Honavar, G. Rudolph, J. Wegener, L. Bull, M. A. Potter, A. C. Schultz, J. F. Miller, E. Burke, & N. Jonoska (Eds.), *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2002)* (pp. 11–18). San Francisco, Morgan Kaufmann.

Bland, R. G., & Shallcross, D. F. (1989). Large traveling salesman problems arising from experiments in X-ray crystallography: A preliminary report on computation. *Operations Research Letters*, *8*, 125–128.

Blum, C. (2002a). ACO applied to group shop scheduling: A case study on intensification and diversification. In M. Dorigo, G. Di Caro, & M. Sampels (Eds.), *Proceedings of ANTS 2002—From Ant Colonies to*

Artificial Ants: Third International Workshop on Ant Algorithms, vol. 2463 of *Lecture Notes in Computer Science* (pp. 14–27). Berlin, Springer-Verlag.

Blum, C. (2002b). *Metaheuristics for Group Shop Scheduling*. DEA thesis, Université Libre de Bruxelles, Brussels.

Blum, C. (2003a). An ant colony optimization algorithm to tackle shop scheduling problems. Technical report TR/IRIDIA/2003-1, IRIDIA, Université Libre de Bruxelles, Brussels.

Blum, C. (2003b). Beam-ACO. Hybridizing ant colony optimization with beam search. An application to open shop scheduling. Technical report TR/IRIDIA/2003-17, IRIDIA, Université Libre de Bruxelles, Brussels.

Blum, C., & Blesa, M. J. (2003). Metaheuristics for the edge-weighted *k*-cardinality tree problem. Technical report LSI-03-1-R, Departament de Llenguatges i Sistemes Informátics, Universitat Politécnica de Catalunya, Barcelona, Spain.

Blum, C., & Dorigo, M. (2003). Deception in ant colony optimization. Part I: Definition and examples. Technical report TR/IRIDIA/2003-18, IRIDIA, Université Libre de Bruxelles, Brussels.

Blum, C., & Dorigo, M. (2004). The hyper-cube framework for ant colony optimization. *IEEE Transactions on Systems, Man, and Cybernetics–Part B*, to appear.

Blum, C., & Roli, A. (2003). Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Computing Surveys*, *35*(3), 268–308.

Blum, C., Roli, A., & Dorigo, M. (2001). HC–ACO: The hyper-cube framework for Ant Colony Optimization. In *Proceedings of MIC'2001—Metaheuristics International Conference*, vol. 2 (pp. 399–403).

Blum, C., & Sampels, M. (2002a). Ant colony optimization for FOP shop scheduling: A case study on different pheromone representations. In D. B. Fogel, M. A. El-Sharkawi, X. Yao, G. Greenwood, H. Iba, P. Marrow, & M. Shackleton (Eds.), *Proceedings of the 2002 Congress on Evolutionary Computation (CEC'02)* (pp. 1558–1563). Piscataway, NJ, IEEE Press.

Blum, C., & Sampels, M. (2002b). When model bias is stronger than selection pressure. In J. J. Merelo, P. Adamidis, H.-G. Beyer, J.-L. Fernández-Villacañas, & H.-P. Schwefel (Eds.), *Proceedings of PPSN-VII, Seventh International Conference on Parallel Problem Solving from Nature*, vol. 2439 in *Lecture Notes in Computer Science* (pp. 893–902). Berlin, Springer-Verlag.

Blum, C., Sampels, M., & Zlochin, M. (2002). On a particularity in model-based search. In W. B. Langdon, E. Cantú-Paz, K. Mathias, R. Roy, D. Davis, R. Poli, K. Balakrishnan, V. Honavar, G. Rudolph, J. Wegener, L. Bull, M. A. Potter, A. C. Schultz, J. F. Miller, E. Burke, & N. Jonoska (Eds.), *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2002)* (pp. 35–42). San Francisco, Morgan Kaufmann.

Boese, K. D., Kahng, A. B., & Muddu, S. (1994). A new adaptive multi-start technique for combinatorial global optimization. *Operations Research Letters*, *16*, 101–113.

Bolondi, M., & Bondanza, M. (1993). Parallelizzazione di un algoritmo per la risoluzione del problema del commesso viaggiatore. Master's thesis, Dipartimento di Elettronica, Politecnico di Milano, Italy.

Bonabeau, E., Dorigo, M., & Theraulaz, G. (1999). *Swarm Intelligence: From Natural to Artificial Systems*. New York, Oxford University Press.

Bonabeau, E., Dorigo, M., & Theraulaz, G. (2000). Inspiration for optimization from social insect behavior. *Nature*, *406*, 39–42.

Bonabeau, E., Henaux, F., Guérin, S., Snyers, D., Kuntz, P., & Theraulaz, G. (1998). Routing in telecommunication networks with "smart" ant-like agents. In S. Albayrak & F. Garijo (Eds.), *Proceedings of IATA'98, Second International Workshop on Intelligent Agents for Telecommunication Applications*, vol. 1437 of *Lecture Notes in Artificial Intelligence* (pp. 60–72). Berlin, Springer-Verlag.

Bonabeau, E., Sobkowski, A., Theraulaz, G., & Deneubourg, J.-L. (1997a). Adaptive task allocation inspired by a model of division of labor in social insects. In D. Lundha, B. Olsson, & A. Narayanan (Eds.), *Bio-Computation and Emergent Computing* (pp. 36–45). Singapore, World Scientific Publishing.

Bonabeau, E., Theraulaz, G., Deneubourg, J.-L., Aron, S., & Camazine, S. (1997b). Self-organization in social insects. *Tree*, *12*(5), 188–193.

Borndörfer, R., Eisenblätter, A., Grötschel, M., & Martin, A. (1998a). The orientation model for frequency assignment problems. Technical report 98-01, Konrad Zuse Zentrum für Informationstechnik, Berlin.

Borndörfer, R., Ferreira, C., & Martin, A. (1998b). Decomposing matrices into blocks. *SIAM Journal on Optimization*, *9*(1), 236–269.

Boyan, J., & Littman, M. (1994). Packet routing in dynamically changing networks: A reinforcement learning approach. In J. Cowan, G. Tesauro, & J. Alspector (Eds.), *Advances in Neural Information Processing Systems 6 (NIPS6)* (pp. 671–678). San Francisco, Morgan Kaufmann.

Branke, J., & Guntsch, M. (2003). New ideas for applying ant colony optimization to the probabilistic TSP. In G. R. Raidl, J.-A. Meyer, M. Middendorf, S. Cagnoni, J. J. R. Cardalda, D. W. Corne, J. Gottlieb, A. Guillot, E. Hart, C. G. Johnson, & E. Marchiori (Eds.), *Applications of Evolutionary Computing, Proceedings of EvoWorkshops 2003*, vol. 2611 of *Lecture Notes in Computer Science* (pp. 165–175). Berlin, Springer-Verlag.

Branke, J., Middendorf, M., & Schneider, F. (1998). Improved heuristics and a genetic algorithm for finding short supersequences. *OR Spektrum*, *20*(1), 39–46.

Bräysy, O. (2003). A reactive variable neighborhood search for the vehicle routing problem with time windows. *INFORMS Journal on Computing*, *15*(4), 347–368.

Brelaz, D. (1979). New methods to color the vertices of a graph. *Communications of the ACM*, *22*, 251–256.

Brixius, N. W., & Anstreicher, K. M. (2001). The Steinberg wiring problem. Technical report, College of Business Administration, University of Iowa, Iowa City.

Brucker, P. (1998). *Scheduling Algorithms*. Berlin, Springer-Verlag.

Brucker, P., Drexl, A., Möhring, R., Neumann, K., & Pesch, E. (1999). Resource-constrained project scheduling: Notation, classification, models, and methods. *European Journal of Operational Research*, *112*(1), 3–41.

Brucker, P., Hurink, J., & Werner, F. (1996). Improving local search heuristics for some scheduling problems—Part I. *Discrete Applied Mathematics*, *65*(1–3), 97–122.

Bruinsma, O. H. (1979). *An Analysis of Building Behaviour of the Termite* Macrotemes subhyalinus. PhD thesis, Lanbouwhogeschool te Wageningen, Netherlands.

Bullnheimer, B., Hartl, R. F., & Strauss, C. (1997). A new rank based version of the Ant System—A computational study. Technical report, Institute of Management Science, University of Vienna, Austria.

Bullnheimer, B., Hartl, R. F., & Strauss, C. (1999a). Applying the Ant System to the vehicle routing problem. In S. Voss, S. Martello, I. H. Osman, & C. Roucairol (Eds.), *Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization* (pp. 285–296). Dordrecht, Netherlands, Kluwer Academic Publishers.

Bullnheimer, B., Hartl, R. F., & Strauss, C. (1999b). An improved ant system algorithm for the vehicle routing problem. *Annals of Operations Research*, *89*, 319–328.

Bullnheimer, B., Hartl, R. F., & Strauss, C. (1999c). A new rank-based version of the Ant System: A computational study. *Central European Journal for Operations Research and Economics*, *7*(1), 25–38.

Bullnheimer, B., Kotsis, G., & Strauss, C. (1998). Parallelization strategies for the Ant System. In R. D. Leone, A. Murli, P. Pardalos, & G. Toraldo (Eds.), *High Performance Algorithms and Software in Nonlinear Optimization*, No. 24 in Kluwer Series of Applied Optmization (pp. 87–100). Dordrecht, Netherlands, Kluwer Academic Publishers.

Burkard, R. E., & Offermann, J. (1977). Entwurf von Schreibmaschinentastaturen mittels quadratischer Zuordnungsprobleme. *Zeitschrift für Operations Research*, *21*, B121–B132.

Camazine, S., Deneubourg, J.-L., Franks, N. R., Sneyd, J., Theraulaz, G., & Bonabeau, E. (Eds.). (2001). *Self-Organization in Biological Systems*. Princeton, NJ, Princeton University Press.

Campos, M., Bonabeau, E., Theraulaz, G., & Deneubourg, J.-L. (2000). Dynamic scheduling and division of labor in social insects. *Adaptive Behavior*, *8*(3), 83–96.

Cantú-Paz, E. (2000). *Efficient and Accurate Parallel Genetic Algorithms*. Boston, Kluwer Academic Publishers.

Caprara, A., Fischetti, M., & Toth, P. (1999). A heuristic method for the set covering problem. *Operations Research*, *47*(5), 730–743.

Casillas, J., Cordón, O., & Herrera, F. (2000). Learning cooperative fuzzy linguistic rules using ant colony algorithms. Technical report DECSAI-00-01-19, Department of Computer Science and Artificial Intelligence, University of Granada, Granada, Spain.

Casillas, J., Cordón, O., & Herrera, F. (2002). COR: A methodology to improve ad hoc data-driven linguistic rule learning methods by inducing cooperation among rules. *IEEE Transactions on Systems, Man, and Cybernetics*, *32*(4), 526–537.

Cerný, V. (1985). A thermodynamical approach to the traveling salesman problem. *Journal of Optimization Theory and Applications*, *45*(1), 41–51.

Chandra, R., Dagum, L., Kohr, D., Maydan, D., McDonald, J., & Menon, R. (2000). *Parallel Programming in OpenMP*. San Francisco, Morgan Kaufmann.

Choi, S., & Yeung, D.-Y. (1996). Predictive Q-routing: A memory-based reinforcement learning approach to adaptive traffic control. In D. Touretzky, M. Mozer, & M. Hasselmo (Eds.), *Advances in Neural Information Processing Systems 8 (NIPS8)* (pp. 945–951). Cambridge, MA, MIT Press.

Chrétien, L. (1996). *Organisation spatiale du matériel provenant de l'excavation du nid chez* Messor barbarus *et des cadavres d'ouvrières chez* Lasius niger *(Hymenopterae: Formicidae)*. PhD thesis, Université Libre de Bruxelles, Brussels.

Christofides, N. (1976). Worst-case analysis of a new heuristic for the travelling salesman problem. Technical report 388, Graduate School of Industrial Administration, Carnegie Mellon University, Pittsburgh.

Cicirello, V. A., & Smith, S. F. (2001). Ant colony control for autonomous decentralized shop floor routing. In *Proceedings of the 5th International Symposium on Autonomous Decentralized Systems* (pp. 383–390). Los Alamitos, CA, IEEE Computer Society Press.

Cicirello, V. A., & Smith, S. F. (2003). Wasp-like agents for distributed factory coordination. *Autonomous Agents and Multi-Agent Systems*, to appear.

Clark, P., & Boswell, R. (1991). Rule induction with CN2: Some recent improvements. In *Proceedings of the European Working Session on Learning (EWSL-91)*, vol. 482 of *Lecture Notes in Artificial Intelligence* (pp. 151–163). Berlin, Springer-Verlag.

Clark, P., & Niblett, T. (1989). The CN2 induction algorithm. *Machine Learning*, *3*(4), 261–283.

Clarke, G., & Wright, J. W. (1964). Scheduling of vehicles from a central depot to a number of delivery points. *Operations Research*, *12*, 568–581.

Coffman, E. G., Jr., Garey, M. R., & Johnson, D. S. (1997). Approximation algorithms for bin packing: A survey. In D. Hochbaum (Ed.), *Approximation Algorithms for NP-Hard Problems* (pp. 46–93). Boston, PWS Publishing.

Colorni, A., Dorigo, M., & Maniezzo, V. (1992a). Distributed optimization by ant colonies. In F. J. Varela & P. Bourgine (Eds.), *Proceedings of the First European Conference on Artificial Life* (pp. 134–142). Cambridge, MA, MIT Press.

Colorni, A., Dorigo, M., & Maniezzo, V. (1992b). An investigation of some properties of an ant algorithm. In R. Männer & B. Manderick (Eds.), *Proceedings of PPSN-II, Second International Conference on Parallel Problem Solving from Nature* (pp. 509–520). Amsterdam, Elsevier.

Colorni, A., Dorigo, M., Maniezzo, V., & Trubian, M. (1994). Ant System for job-shop scheduling. *JORBEL—Belgian Journal of Operations Research, Statistics and Computer Science*, *34*(1), 39–53.

Congram, R. K., Potts, C. N., & de Velde, S. L. V. (2002). An iterated dynasearch algorithm for the single–machine total weighted tardiness scheduling problem. *INFORMS Journal on Computing*, *14*(1), 52–67.

Cook, W. J., Cunningham, W. H., Pulleyblank, W. R., & Schrijver, A. (1998). *Combinatorial Optimization*. New York, John Wiley & Sons.

Cooper, G. F., & Herskovits, E. (1992). A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, *9*(4), 309–348.

Cordón, O., de Viana, I. F., & Herrera, F. (2002). Analysis of the best-worst Ant System and its variants on the TSP. *Mathware and Soft Computing*, *9*(2–3), 177–192.

Cordón, O., de Viana, I. F., Herrera, F., & Moreno, L. (2000). A new ACO model integrating evolutionary computation concepts: The best-worst Ant System. In M. Dorigo, M. Middendorf, & T. Stützle (Eds.), *Abstract Proceedings of ANTS 2000—From Ant Colonies to Artificial Ants: Second International Workshop on Ant Algorithms* (pp. 22–29). Brussels, IRIDIA, Université Libre de Bruxelles.

Cordón, O., & Herrera, F. (2000). A proposal for improving the accuracy of linguistic modeling. *IEEE Transactions on Fuzzy Systems*, *8*(3), 335–344.

Cordone, R., & Maffioli, F. (2001). Coloured Ant System and local search to design local telecommunication networks. In E. J. W. Boers, J. Gottlieb, P. L. Lanzi, R. E. Smith, S. Cagnoni, E. Hart, G. R. Raidl, & H. Tijink (Eds.), *Applications of Evolutionary Computing: Proceedings of EvoWorkshops 2001*, vol. 2037 of *Lecture Notes in Computer Science* (pp. 60–69). Berlin, Springer-Verlag.

Cordone, R., & Maffioli, F. (2003). On the complexity of graph tree partition problems. *Discrete Applied Mathematics*, *134*(1–3), 51–65.

Corne, D., Dorigo, M., & Glover, F. (Eds.). (1999). *New Ideas in Optimization*. London, McGraw Hill.

Costa, D., & Hertz, A. (1997). Ants can colour graphs. *Journal of the Operational Research Society*, *48*, 295–305.

Crauwels, H. A. J., Potts, C. N., & Wassenhove, L. N. V. (1998). Local search heuristics for the single machine total weighted tardiness scheduling problem. *INFORMS Journal on Computing*, *10*(3), 341–350.

Crescenzi, P., Goldman, D., Papadimitriou, C. H., Piccolboni, A., & Yannakakis, M. (1998). On the complexity of protein folding. *Journal of Computational Biology*, *5*(3), 423–466.

Croes, G. A. (1958). A method for solving traveling salesman problems. *Operations Research*, *6*, 791–812.

Czyzak, P., & Jaszkiewicz, A. (1998). Pareto simulated annealing—A metaheuristic technique for multiple objective combinatorial optimization. *Journal of Multi-Criteria Decision Analysis*, *7*, 34–47.

Dantzig, G. B., Fulkerson, D. R., & Johnson, S. M. (1954). Solution of a large-scale traveling salesman problem. *Operations Research*, *2*, 393–410.

Davenport, A., Tsang, E., Wang, C. J., & Zhu, K. (1994). GENET: A connectionist architecture for solving constraint satisfaction problems by iterative improvement. In *Proceedings of the 14th National Conference on Artificial Intelligence* (pp. 325–330). Menlo Park, CA, AAAI Press/MIT Press.

Dawid, H., Doerner, K., Hartl, R. F., & Reimann, M. (2002). Ant systems to solve operational problems. In H. Dawid, K. Doerner, G. Dorffner, T. Fent, M. Feurstein, R. F. Hartl, A. Mild, M. Natter, M. Reimann, & A. Taudes (Eds.), *Quantitative Models of Learning Organizations* (pp. 65–82). Vienna, Springer-Verlag.

De Bonet, J. S., Isbell, C. L., & Viola, P. (1997). MIMIC: Finding optima by estimating probability densities. In M. C. Mozer, M. I. Jordan, & T. Petsche (Eds.), *Advances in Neural Information Processing Systems 9 (NIPS9)*, vol. 9 (pp. 424–431). Cambridge, MA, MIT Press.

de Campos, L. M., Fernández-Luna, J. M., Gámez, J. A., & Puerta, J. M. (2002a). Ant colony optimization for learning Bayesian networks. *International Journal of Approximate Reasoning*, *31*(3), 291–311.

de Campos, L. M., Gámez, J. A., & Puerta, J. M. (2002b). Learning Bayesian networks by ant colony optimisation: Searching in the space of orderings. *Mathware and Soft Computing*, *9*(2–3), 251–268.

de Campos, L. M., & Puerta, J. M. (2001). Stochastic local search and distributed search algorithms for learning Bayesian networks. In *III International Symposium on Adaptive Systems (ISAS): Evolutionary Computation and Probabilisitic Graphical Models* (pp. 109–115). La Habana, Cuba: Institute of Cybernetics, Mathematics and Physics.

Deb, K. (2001). *Multi-Objective Optimization Using Evolutionary Algorithms*. Chichester, UK, John Wiley & Sons.

Dechter, R. (2003). *Constraint Processing*. San Francisco, Morgan Kaufmann.

Dechter, R., Meiri, I., & Pearl, J. (1991). Temporal constraint networks. *Artificial Intelligence*, *49*(1–3), 61–95.

Dechter, R., & Pearl, J. (1989). Tree clustering schemes for constraint-processing. *Artificial Intelligence*, *38*(3), 353–366.

Delisle, P., Krajecki, M., Gravel, M., & Gagné, C. (2001). Parallel implementation of an ant colony optimization metaheuristic with OpenMP. In *Proceedings of the 3rd European Workshop on OpenMP (EWOMP'01)*, Barcelona, Spain.

Dell'Amico, M., Maffioli, F., & Martello, S. (Eds.). (1997). *Annotated Bibliographies in Combinatorial Optimization*. Chichester, UK, John Wiley & Sons.

den Besten, M. (2000). Ants for the single machine total weighted tardiness problem. Master's thesis, University of Amsterdam.

den Besten, M. L., Stützle, T., & Dorigo, M. (2000). Ant colony optimization for the total weighted tardiness problem. In M. Schoenauer, K. Deb, G. Rudolph, X. Yao, E. Lutton, J. J. Merelo, & H.-P. Schwefel (Eds.), *Proceedings of PPSN-VI, Sixth International Conference on Parallel Problem Solving from Nature*, vol. 1917 of *Lecture Notes in Computer Science* (pp. 611–620). Berlin, Springer-Verlag.

Deneubourg, J.-L. (2002). Personal communication. Université Libre de Bruxelles, Brussels.

Deneubourg, J.-L., Aron, S., Goss, S., & Pasteels, J.-M. (1990). The self-organizing exploratory pattern of the Argentine ant. *Journal of Insect Behavior*, *3*, 159–168.

Deneubourg, J.-L., Goss, S., Franks, N., Sendova-Franks, A., Detrain, C., & Chrétien, L. (1991). The dynamics of collective sorting: Robot-like ants and ant-like robots. In J.-A. Meyer & S. W. Wilson (Eds.), *Proceedings of the First International Conference on Simulation of Adaptive Behavior: From Animals to Animats* (pp. 356–363). Cambridge, MA, MIT Press.

Di Caro, G. (in preparation). *Systems of Ant-like Agents for Adaptive Network Control and Combinatorial Optimization*. PhD thesis, Université Libre de Bruxelles, Brussels.

Di Caro, G., & Dorigo, M. (1997). AntNet: A mobile agents approach to adaptive routing. Technical report IRIDIA/97-12, IRIDIA, Université Libre de Bruxelles, Brussels.

Di Caro, G., & Dorigo, M. (1998a). An adaptive multi-agent routing algorithm inspired by ants behavior. In K. A. Hawick & H. A. James (Eds.), *Proceedings of PART98—5th Annual Australasian Conference on Parallel and Real-Time Systems* (pp. 261–272). Singapore, Springer-Verlag.

Di Caro, G., & Dorigo, M. (1998b). Ant colonies for adaptive routing in packet-switched communications networks. In A. E. Eiben, T. Bäck, M. Schoenauer, & H.-P. Schwefel (Eds.), *Proceedings of PPSN-V, Fifth International Conference on Parallel Problem Solving from Nature*, vol. 1498 of *Lecture Notes in Computer Science* (pp. 673–682). Berlin, Springer-Verlag.

Di Caro, G., & Dorigo, M. (1998c). AntNet: Distributed stigmergetic control for communications networks. *Journal of Artificial Intelligence Research*, *9*, 317–365.

Di Caro, G., & Dorigo, M. (1998d). Extending AntNet for best-effort quality-of-service routing. Unpublished presentation at *ANTS'98—From Ant Colonies to Artificial Ants: First International Workshop on Ant Colony Optimization*, Brussels.

Di Caro, G., & Dorigo, M. (1998e). Mobile agents for adaptive routing. In H. El-Rewini (Ed.), *Proceedings of the 31st International Conference on System Sciences (HICSS-31)* (pp. 74–83). Los Alamitos, CA, IEEE Computer Society Press.

Di Caro, G., & Dorigo, M. (1998f). Two ant colony algorithms for best-effort routing in datagram networks. In Y. Pan, S. G. Akl, & K. Li (Eds.), *Proceedings of the Tenth IASTED International Conference on Parallel and Distributed Computing and Systems (PDCS'98)* (pp. 541–546). Anaheim, CA, IASTED/ACTA Press.

Dickey, J. W., & Hopkins, J. W. (1972). Campus building arrangement using TOPAZ. *Transportation Science*, *6*, 59–68.

Dijkstra, E. W. (1959). A note on two problems in connection with graphs. *Numerische Mathematik*, *1*, 269–271.

Doerner, K., Gutjahr, W. J., Hartl, R. F., Strauss, C., & Stummer, C. (2003). Pareto ant colony optimization: A metaheuristic approach to multiobjective portfolio selection. *Annals of Operations Research*, to appear.

Doerner, K., Hartl, R. F., & Reimann, M. (2001). Cooperative ant colonies for optimizing resource allocation in transportation. In E. J. W. Boers, J. Gottlieb, P. L. Lanzi, R. E. Smith, S. Cagnoni, E. Hart, G. R. Raidl, & H. Tijink (Eds.), *Applications of Evolutionary Computing: Proceedings of EvoWorkshops 2001*, vol. 2037 of *Lecture Notes in Computer Science* (pp. 70–79). Berlin, Springer-Verlag.

Doerner, K., Hartl, R. F., & Reimann, M. (2003). Competants for problem solving: The case of full truckload transportation. *Central European Journal for Operations Research and Economics*, *11*(2), 115–141.

Dorigo, M. (1992). *Optimization, Learning and Natural Algorithms* [in Italian]. PhD thesis, Dipartimento di Elettronica, Politecnico di Milano, Milan.

Dorigo, M. (2001). Ant algorithms solve difficult optimization problems. In J. Kelemen (Ed.), *Proceedings of the Sixth European Conference on Artificial Life*, vol. 2159 of *Lecture Notes in Artificial Intelligence* (pp. 11–22). Berlin, Springer-Verlag.

Dorigo, M., Bonabeau, E., & Theraulaz, G. (2000a). Ant algorithms and stigmergy. *Future Generation Computer Systems*, *16*(8), 851–871.

Dorigo, M., & Di Caro, G. (1999a). Ant colony optimization: A new meta-heuristic. In P. J. Angeline, Z. Michalewicz, M. Schoenauer, X. Yao, & A. Zalzala (Eds.), *Proceedings of the 1999 Congress on Evolutionary Computation (CEC'99)* (pp. 1470–1477). Piscataway, NJ, IEEE Press.

Dorigo, M., & Di Caro, G. (1999b). The ant colony optimization meta-heuristic. In D. Corne, M. Dorigo, & F. Glover (Eds.), *New Ideas in Optimization* (pp. 11–32). London, McGraw Hill.

Dorigo, M., Di Caro, G., & Gambardella, L. M. (1999). Ant algorithms for discrete optimization. *Artificial Life*, *5*(2), 137–172.

Dorigo, M., Di Caro, G., & Sampels, M. (Eds.). (2002a). *Proceedings of ANTS 2002—From Ant Colonies to Artificial Ants: Third International Workshop on Ant Algorithms*, vol. 2463 of *Lecture Notes in Computer Science*. Berlin, Springer-Verlag.

Dorigo, M., & Gambardella, L. M. (1996). A study of some properties of Ant-Q. In H. Voigt, W. Ebeling, I. Rechenberg, & H. Schwefel (Eds.), *Proceedings of PPSN-IV, Fourth International Conference on Parallel Problem Solving from Nature*, vol. 1141 of *Lecture Notes in Computer Science* (pp. 656–665). Berlin, Springer-Verlag.

Dorigo, M., & Gambardella, L. M. (1997a). Ant colonies for the traveling salesman problem. *BioSystems*, *43*(2), 73–81.

Dorigo, M., & Gambardella, L. M. (1997b). Ant Colony System: A cooperative learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computation*, *1*(1), 53–66.

Dorigo, M., Gambardella, L. M., Middendorf, M., & Stützle, T. (Eds.). (2002b). Special section on "Ant Colony Optimization." *IEEE Transactions on Evolutionary Computation*, *6*(4), 317–365.

Dorigo, M., Maniezzo, V., & Colorni, A. (1991a). Positive feedback as a search strategy. Technical report 91-016, Dipartimento di Elettronica, Politecnico di Milano, Milan.

Dorigo, M., Maniezzo, V., & Colorni, A. (1991b). The Ant System: An autocatalytic optimizing process. Technical report 91-016 revised, Dipartimento di Elettronica, Politecnico di Milano, Milan.

Dorigo, M., Maniezzo, V., & Colorni, A. (1996). Ant System: Optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics—Part B*, *26*(1), 29–41.

Dorigo, M., Middendorf, M., & Stützle, T. (Eds.). (2000b). *Abstract Proceedings of ANTS 2000—From Ant Colonies to Artificial Ants: Second International Workshop on Ant Algorithms*. Brussels, IRIDIA, Université Libre de Bruxelles.

Dorigo, M., & Stützle, T. (2001). An experimental study of the simple ant colony optimization algorithm. In N. Mastorakis (Ed.), *2001 WSES International Conference on Evolutionary Computation (EC'01)* (pp. 253–258). WSES Press.

Dorigo, M., & Stützle, T. (2002). The ant colony optimization metaheuristic: Algorithms, applications and advances. In F. Glover & G. Kochenberger (Eds.), *Handbook of Metaheuristics*, vol. 57 of *International Series in Operations Research & Management Science* (pp. 251–285). Norwell, MA, Kluwer Academic Publishers.

Dorigo, M., Stützle, T., & Di Caro, G. (Eds.). (2000c). Special issue on "Ant Algorithms." *Future Generation Computer Systems*, *16*, 851–956.

Dorigo, M., Trianni, V., Şahin, E., Labella, T., Gross, R., Baldassarre, G., Nolfi, S., Deneubourg, J.-L., Mondada, F., Floreano, D., & Gambardella, L. M. (2003). Evolving self-organizing behaviors for a *Swarm-bot*. Technical report IRIDIA/2003-11, IRIDIA, Université Libre de Bruxelles, Brussels.

Dorigo, M., Zlochin, M., Meuleau, N., & Birattari, M. (2002c). Updating ACO pheromones using stochastic gradient ascent and cross-entropy methods. In S. Cagnoni, J. Gottlieb, E. Hart, M. Middendorf, & G. R. Raidl (Eds.), *Applications of Evolutionary Computing, Proceedings of EvoWorkshops 2002*, vol. 2279 of *Lecture Notes in Computer Science* (pp. 21–30). Berlin, Springer-Verlag.

Dorne, R., & Hao, J. (1999). Tabu search for graph coloring, t-colorings and set t-colorings. In S. Voss, S. Martello, I. Osman, & C. Roucairol (Eds.), *Meta-heuristics: Advances and Trends in Local Search Paradigms for Optimization* (pp. 77–92). Boston, Kluwer Academic Publishers.

Elmaghraby, S. E. (1977). *Activity Networks*. New York, John Wiley & Sons.

Elshafei, A. N. (1977). Hospital layout as a quadratic assignment problem. *Operations Research Quarterly*, *28*, 167–179.

Eyckelhof, C. J., & Snoek, M. (2002). Ant systems for a dynamic TSP: Ants caught in a traffic jam. In M. Dorigo, G. Di Caro, & M. Sampels (Eds.), *Proceedings of ANTS 2002—From Ant Colonies to Artificial Ants: Third International Workshop on Ant Algorithms*, vol. 2463 of *Lecture Notes in Computer Science* (pp. 88–99). Berlin, Springer-Verlag.

Fabrikant, S. I. (2000). *Spatial Metaphors for Browsing Large Data Archives*. PhD thesis, Department of Geography, University of Colorado at Boulder.

Faigle, U., & Kern, W. (1992). Some convergence results for probabilistic tabu search. *ORSA Journal on Computing*, *4*(1), 32–37.

Falkenauer, E. (1996). A hybrid grouping genetic algorithm for bin packing. *Journal of Heuristics*, *2*(1), 5–30.

Fang, H.-L., Ross, P., & Corne, D. (1994). A promising hybrid GA/heuristic approach for open-shop scheduling problems. In A. G. Cohn (Ed.), *Proceedings of the 11th European Conference on Artificial Intelligence* (pp. 590–594). Chichester, John Wiley & Sons.

Fenet, S., & Solnon, C. (2003). Searching for maximum cliques with ant colony optimization. In G. R. Raidl, J.-A. Meyer, M. Middendorf, S. Cagnoni, J. J. R. Cardalda, D. W. Corne, J. Gottlieb, A. Guillot, E. Hart, C. G. Johnson, & E. Marchiori (Eds.), *Applications of Evolutionary Computing, Proceedings of EvoWorkshops 2003*, vol. 2611 of *Lecture Notes in Computer Science* (pp. 236–245). Berlin, Springer-Verlag.

Feo, T. A., & Resende, M. G. C. (1989). A probabilistic heuristic for a computationally difficult set covering problem. *Operations Research Letters*, *8*, 67–71.

Feo, T. A., & Resende, M. G. C. (1995). Greedy randomized adaptive search procedures. *Journal of Global Optimization*, *6*, 109–133.

Festa, P., & Resende, M. G. C. (2002). GRASP: An annotated bibliography. In P. Hansen & C. C. Ribeiro (Eds.), *Essays and Surveys on Metaheuristics* (pp. 325–367). Boston, Kluwer Academic Publishers.

Fischetti, M., Hamacher, H. W., Jörnsten, K., & Maffioli, F. (1994). Weighted *k*-cardinality trees: Complexity and polyhedral structure. *Networks*, *24*, 11–21.

Fleurent, C., & Ferland, J. A. (1996). Genetic and hybrid algorithms for graph coloring. *Annals of Operations Research*, *63*, 437–461.

Flood, M. M. (1956). The traveling-salesman problem. *Operations Research*, *4*, 61–75.

Fogel, D. B. (1995). *Evolutionary Computation*. Piscataway, NJ, IEEE Press.

Fogel, L. J., Owens, A. J., & Walsh, M. J. (1966). *Artificial Intelligence through Simulated Evolution*. New York, John Wiley & Sons.

Ford, L., & Fulkerson, D. (1962). *Flows in Networks*. Princeton, NJ, Princeton University Press.

Foulds, L., Hamacher, H., & Wilson, J. (1998). Integer programming approaches to facilities layout models with forbidden areas. *Annals of Operations Research*, *81*, 405–417.

Foulser, D. E., Li, M., & Yang, Q. (1992). Theory and algorithms for plan merging. *Artificial Intelligence*, *57*(2–3), 143–181.

Frank, J. (1996). Weighting for Godot: Learning heuristics for GSAT. In *Proceedings of the AAAI National Conference on Artificial Intelligence* (pp. 338–343). Menlo Park, CA, AAAI Press/MIT Press.

Franks, N. R. (1986). Teams in social insects: Group retrieval of prey by army ants (*Eciton burchelli*, Hymenoptera: Formicidae). *Behavioral Ecology and Sociobiology*, *18*, 425–429.

Franks, N. R., & Sendova-Franks, A. B. (1992). Brood sorting by ants: Distributing the workload over the work surface. *Behavioral Ecology and Sociobiology*, *30*, 109–123.

Freuder, E. C., & Wallace, R. J. (1992). Partial constraint satisfaction. *Artificial Intelligence*, *58*(1–3), 21–70.

Fujita, K., Saito, A., Matsui, T., & Matsuo, H. (2002). An adaptive ant-based routing algorithm used routing history in dynamic networks. In L. Wang, K. C. T. Furuhashi, J.-H. Kim, & X. Yao (Eds.), *4th Asia-Pacific Conference on Simulated Evolution and Learning (SEAL'02)*, vol. 1 (pp. 46–50). Orchid Country Club, Singapore, 18–22 Nov. 2002.

Gagné, C., Price, W. L., & Gravel, M. (2002). Comparing an ACO algorithm with other heuristics for the single machine scheduling problem with sequence-dependent setup times. *Journal of the Operational Research Society*, *53*, 895–906.

Galinier, P., & Hao, J.-K. (1997). Tabu search for maximal constraint satisfaction problems. In G. Smolka (Ed.), *Principles and Practice of Constraint Programming—CP97*, vol. 1330 of *Lecture Notes in Computer Science* (pp. 196–208). Berlin, Springer-Verlag.

Galinier, P., & Hao, J.-K. (1999). Hybrid evolutionary algorithms for graph coloring. *Journal of Combinatorial Optimization*, *3*(4), 379–397.

Gambardella, L. M., & Dorigo, M. (1995). Ant-Q: A reinforcement learning approach to the traveling salesman problem. In A. Prieditis & S. Russell (Eds.), *Proceedings of the Twelfth International Conference on Machine Learning (ML-95)* (pp. 252–260). Palo Alto, CA, Morgan Kaufmann.

Gambardella, L. M., & Dorigo, M. (1996). Solving symmetric and asymmetric TSPs by ant colonies. In T. Baeck, T. Fukuda, & Z. Michalewicz (Eds.), *Proceedings of the 1996 IEEE International Conference on Evolutionary Computation (ICEC'96)* (pp. 622–627). Piscataway, NJ, IEEE Press.

Gambardella, L. M., & Dorigo, M. (1997). HAS-SOP: An hybrid Ant System for the sequential ordering problem. Technical report IDSIA-11-97, IDSIA, Lugano, Switzerland.

Gambardella, L. M., & Dorigo, M. (2000). Ant Colony System hybridized with a new local search for the sequential ordering problem. *INFORMS Journal on Computing*, *12*(3), 237–255.

Gambardella, L. M., Taillard, É. D., & Agazzi, G. (1999a). MACS-VRPTW: A multiple ant colony system for vehicle routing problems with time windows. In D. Corne, M. Dorigo, & F. Glover (Eds.), *New Ideas in Optimization* (pp. 63–76). London, McGraw Hill.

Gambardella, L. M., Taillard, E. D., & Dorigo, M. (1999b). Ant colonies for the quadratic assignment problem. *Journal of the Operational Research Society*, *50*(2), 167–176.

Gámez, J. A., & Puerta, J. M. (2002). Searching the best elimination sequence in Bayesian networks by using ant colony optimization. *Pattern Recognition Letters*, *23*(1–3), 261–277.

Gamst, A. (1986). Some lower bounds for a class of frequency assignment problems. *IEEE Transactions of Vehicular Technology*, *35*(1), 8–14.

Garey, M. R., & Johnson, D. S. (1979). *Computers and Intractability: A Guide to the Theory of $\mathcal{NP}$-Completeness*. San Francisco, Freeman.

Geman, S., & Geman, D. (1984). Stochastic relaxation, Gibbs distribution, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *6*, 721–741.

Giffler, B., & Thompson, G. L. (1960). Algorithms for solving production scheduling problems. *Operations Research*, *8*, 487–503.

Gilmore, P. C. (1962). Optimal and suboptimal algorithms for the quadratic assignment problem. *Journal of the SIAM*, *10*, 305–313.

Glover, F. (1977). Heuristics for integer programming using surrogate constraints. *Decision Sciences*, *8*, 156–166.

Glover, F. (1989). Tabu search—Part I. *ORSA Journal on Computing*, *1*(3), 190–206.

Glover, F. (1990). Tabu search—Part II. *ORSA Journal on Computing*, *2*(1), 4–32.

Glover, F. (1996). Ejection chains, reference structures and alternating path methods for traveling salesman problems. *Discrete Applied Mathematics*, *65*(1–3), 223–253.

Glover, F., & Hanafi, S. (2002). Tabu search and finite convergence. *Discrete Applied Mathematics*, *119*(1–2), 3–36.

Glover, F., & Kochenberger, G. (Eds.). (2002). *Handbook of Metaheuristics*. Norwell, MA, Kluwer Academic Publishers.

Glover, F., & Laguna, M. (1997). *Tabu Search*. Boston, Kluwer Academic Publishers.

Glover, F., Laguna, M., & Martí, R. (2002). Scatter search and path relinking: Advances and applications. In F. Glover & G. Kochenberger (Eds.), *Handbook of Metaheuristics*, vol. 57 of *International Series in Operations Research & Management Science* (pp. 1–35). Norwell, MA, Kluwer Academic Publishers.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*. Reading, MA, Addison-Wesley.

Golden, B. L., & Stewart, W. R. (1985). Enpirical analysis of heuristics. In E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan, & D. B. Shmoys (Eds.), *The Traveling Salesman Problem* (pp. 307–360). Chichester, UK, John Wiley & Sons.

Goss, S., Aron, S., Deneubourg, J. L., & Pasteels, J. M. (1989). Self-organized shortcuts in the Argentine ant. *Naturwissenschaften*, *76*, 579–581.

Gottlieb, J., Puchta, M., & Solnon, C. (2003). A study of greedy, local search, and ant colony optimization approaches for car sequencing problems. In G. R. Raidl, J.-A. Meyer, M. Middendorf, S. Cagnoni, J. J. R. Cardalda, D. W. Corne, J. Gottlieb, A. Guillot, E. Hart, C. G. Johnson, & E. Marchiori (Eds.), *Applications of Evolutionary Computing, Proceedings of EvoWorkshops 2003*, vol. 2611 of *Lecture Notes in Computer Science* (pp. 246–257). Berlin, Springer-Verlag.

Grabowski, J., & Wodecki, M. (2001). A new very fast tabu search algorithm for the job shop problem. Technical report 21/2001, Wroclaw University of Technology, Institute of Engineering Cybernetics, Wroclaw, Poland.

Grassé, P. P. (1959). La reconstruction du nid et les coordinations interindividuelles chez *Bellicositermes natalensis* et *Cubitermes* sp. La théorie de la stigmergie: Essai d'interprétation du comportement des termites constructeurs. *Insectes Sociaux*, *6*, 41–81.

Gravel, M., Price, W. L., & Gagné, C. (2002). Scheduling continuous casting of aluminum using a multiple objective ant colony optimization metaheuristic. *European Journal of Operational Research*, *143*(1), 218–229.

Grosso, A., Della Croce, F., & Tadei, R. (2004). An enhanced dynasearch neighborhood for the single-machine total weighted tardiness scheduling problem. *Operations Research Letters*, *32*(1), 68–72.

Grötschel, M. (1981). On the symmetric travelling salesman problem: Solution of a 120-city problem. *Mathematical Programming Study*, *12*, 61–77.

Grötschel, M., & Holland, O. (1991). Solution of large-scale symmetric traveling salesman problems. *Mathematical Programming*, *51*, 141–202.

Guesgen, H., & Hertzberg, J. (1992). *A Perspective of Constraint-Based Reasoning*, vol. 597 of *Lecture Notes in Artificial Intelligence*. Berlin, Springer-Verlag.

Güneş, M., Sorges, U., & Bouazizi, I. (2002). ARA—The ant-colony based routing algorithm for MANETS. In S. Olariu (Ed.), *2002 ICPP Workshop on Ad Hoc Networks (IWAHN 2002)* (pp. 79–85). Los Alamitos, CA, IEEE Computer Society Press.

Güneş, M., & Spaniol, O. (2002). Routing algorithms for mobile multi-hop ad-hoc networks. In H. Turlakov & L. Boyanov (Eds.), *International Workshop on Next Generation Network Technologies* (pp. 10–24). Rousse, Bulgaria: Central Laboratory for Parallel Processing—Bulgarian Academy of Sciences.

Guntsch, M., & Middendorf, M. (2001). Pheromone modification strategies for ant algorithms applied to dynamic TSP. In E. J. W. Boers, J. Gottlieb, P. L. Lanzi, R. E. Smith, S. Cagnoni, E. Hart, G. R. Raidl, & H. Tijink (Eds.), *Applications of Evolutionary Computing*, vol. 2037 of *Lecture Notes in Computer Science* (pp. 213–222). Berlin, Springer-Verlag.

Guntsch, M., & Middendorf, M. (2002a). Applying population based ACO to dynamic optimization problems. In M. Dorigo, G. Di Caro, & M. Sampels (Eds.), *Proceedings of ANTS 2002—From Ant Colonies to Artificial Ants: Third International Workshop on Ant Algorithms*, vol. 2463 of *Lecture Notes in Computer Science* (pp. 111–122). Berlin, Springer-Verlag.

Guntsch, M., & Middendorf, M. (2002b). A population based approach for ACO. In S. Cagnoni, J. Gottlieb, E. Hart, M. Middendorf, & G. R. Raidl (Eds.), *Applications of Evolutionary Computing*, vol. 2279 of *Lecture Notes in Computer Science* (pp. 71–80). Berlin, Springer-Verlag.

Guntsch, M., Middendorf, M., & Schmeck, H. (2001). An ant colony optimization approach to dynamic TSP. In L. Spector, E. D. Goodman, A. Wu, W. B. Langdon, H.-M. Voigt, M. Gen, S. Sen, M. Dorigo, S. Pezeshk, M. H. Garzon, & E. Burke (Eds.), *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)* (pp. 860–867). San Francisco, Morgan Kaufmann.

Guntsch, M. G., & Middendorf, M. (2003). Solving multi-criteria optimization problems with population-based ACO. In C. M. Fonseca, P. J. Fleming, E. Zitzler, K. Deb, & L. Thiele (Eds.), *Evolutionary Multi-Criterion Optimization*, vol. 2632 of *Lecture Notes in Computer Science* (pp. 464–478). Berlin, Springer-Verlag.

Gutjahr, W. J. (2000). A graph-based Ant System and its convergence. *Future Generation Computer Systems*, *16*(8), 873–888.

Gutjahr, W. J. (2002). ACO algorithms with guaranteed convergence to the optimal solution. *Information Processing Letters*, *82*(3), 145–153.

Hadji, R., Rahoual, M., Talbi, E., & Bachelet, V. (2000). Ant colonies for the set covering problem. In M. Dorigo, M. Middendorf, & T. Stützle (Eds.), *Abstract Proceedings of ANTS 2000—From Ant Colonies to Artificial Ants: Second International Workshop on Ant Algorithms* (pp. 63–66). Brussels, Université Libre de Bruxelles.

Hahn, P., & Krarup, J. (2001). A hospital facility layout problem finally solved. *Journal of Intelligent Manufacturing*, *12*(5–6), 487–496.

Hahn, P. M., Hightower, W. L., Johnson, T. A., Guignard-Spielberg, M., & Roucairol, C. (2001). Tree elaboration strategies in branch and bound algorithms for solving the quadratic assignment problem. *Yugoslavian Journal of Operational Research*, *11*(1), 41–60.

Hajek, B. (1988). Cooling schedules for optimal annealing. *Mathematics of Operations Research*, *13*(2), 311–329.

Haken, H. (1983). *Synergetics*. Berlin, Springer-Verlag.

Hamacher, H. W., & Jörnsten, K. (1993). Optimal relinquishment according to the Norwegian petrol law: A combinatorial optimization approach. Technical report 7/93, Norwegian School of Economics and Business Administration, Bergen, Norway.

Hanafi, S. (2000). On the convergence of tabu search. *Journal of Heuristics*, *7*(1), 47–58.

Handl, J., & Meyer, B. (2002). Improved ant-based clustering and sorting in a document retrieval interface. In J. J. Merelo, P. Adamidis, H.-G. Beyer, J.-L. Fernández-Villacañas, & H.-P. Schwefel (Eds.), *Proceedings of PPSN-VII, Seventh International Conference on Parallel Problem Solving from Nature*, vol. 2439 in *Lecture Notes in Computer Science* (pp. 913–923). Berlin, Springer-Verlag.

Hansen, P., & Mladenović, N. (1999). An introduction to variable neighborhood search. In S. Voss, S. Martello, I. H. Osman, & C. Roucairol (Eds.), *Meta-Heuristics—Advances and Trends in Local Search Paradigms for Optimization* (pp. 433–458). Dordrecht, Netherlands, Kluwer Academic Publishers.

Hansen, P., & Ribeiro, C. (Eds.). (2001). *Essays and Surveys on Metaheuristics*. Boston, Kluwer Academic Publishers.

Hartmann, S., & Kolisch, R. (1999). Self adapting genetic algorithm with an application to project scheduling. Technical report 506, University of Kiel, Kiel, Germany.

Hartmann, S., & Kolisch, R. (2000). Experimental evaluation of state-of-the-art heuristics for resource constrained project scheduling. *European Journal of Operational Research*, *127*(2), 394–407.

Haupt, R. (1989). A survey of priority rule-based scheduling. *OR Spektrum*, *11*, 3–6.

Heckerman, D., Geiger, D., & Chickering, D. M. (1995). Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, *20*(3), 197–244.

Heissenbüttel, M., & Braun, T. (2003). Ants-based routing in large-scale mobile ad-hoc networks. In K. Irmscher & R.-P. Fähnrich (Eds.), *Proceedings of Kommunikation in verteilten Systemen (KiVS '03)* (pp. 91–99). Berlin, VDE Verlag GmbH.

Helsgaun, K. (2000). An effective implementation of the Lin-Kernighan traveling salesman heuristic. *European Journal of Operational Research*, *126*(1), 106–130.

Hertz, A., Taillard, É. D., & de Werra, D. (1997). A tutorial on tabu search. In E. H. L. Aarts & J. K. Lenstra (Eds.), *Local Search in Combinatorial Optimization* (pp. 121–136). Chichester, UK, John Wiley & Sons.

Heusse, M. (2001). *Routage et équilibrage de charge par agents dans les réseaux de communication*. PhD thesis, École des Hautes Études en Sciences Sociales, Paris.

Heusse, M., & Kermarrec, Y. (2000). Adaptive routing and load balancing of ephemeral connections. In *Proceedings of the 1st IEEE European Conference on Universal Multiservice Networks ECUMN'2000* (pp. 100–108). Piscataway, NJ, IEEE Press.

Heusse, M., Snyers, D., Guérin, S., & Kuntz, P. (1998). Adaptive agent-driven routing and load balancing in communication networks. *Advances in Complex Systems*, *1*(2), 237–254.

Hochbaum, D. S. (Ed.). (1997). *Approximation Algorithms for NP-Hard Problems*. Boston, PWS Publishing Company.

Holland, J. (1975). *Adaptation in Natural and Artificial Systems*. Ann Arbor, University of Michigan Press.

Holland, O., & Melhuish, C. (1999). Stigmergy, self-organization, and sorting in collective robotics. *Artificial Life*, *5*(2), 173–202.

Hromkovic, J. (2003). *Algorithmics for Hard Problems*, 2nd ed. Berlin, Springer-Verlag.

Hsu, H.-P., Mehra, V., Nadler, W., & Grassberger, P. (2003). Growth algorithms for lattice heteropolymers at low temperatures. *Journal of Chemical Physics*, *118*(1), 444–451.

Hurkens, C. A. J., & Tiourine, S. R. (1995). Upper and lower bounding techniques for frequency assignment problems. Technical report 95-34, Department of Mathematics and Computing Science, Eindhoven University of Technology, Netherlands.

Iredi, S., Merkle, D., & Middendorf, M. (2001). Bi-criterion optimization with multi colony ant algorithms. In E. Zitzler, K. Deb, L. Thiele, C. C. Coello, & D. Corne (Eds.), *First International Conference on*

Evolutionary Multi-Criterion Optimization (EMO'01), vol. 1993 of *Lecture Notes in Computer Science* (pp. 359–372). Berlin, Springer-Verlag.

Jacobs, L. W., & Brusco, M. J. (1995). A local search heuristic for large set covering problems. *Naval Research Logistics*, *42*, 1129–1140.

Jaillet, P. (1985). *Probabilistic Traveling Salesman Problems*. PhD thesis, MIT, Cambridge, MA.

Jaillet, P. (1988). A priori solution of a travelling salesman problem in which a random subset of the customers are visited. *Operations Research*, *36*(6), 929–936.

Jensen, F. V. (2001). *Bayesian Networks and Decision Graphs*. Berlin, Springer-Verlag.

Johnson, D. S., Aragon, C. R., McGeoch, L. A., & Schevon, C. (1991). Optimization by simulated annealing: An experimental evaluation: Part II, Graph coloring and number partitioning. *Operations Research*, *39*(3), 378–406.

Johnson, D. S., Gutin, G., McGeoch, L. A., Yeo, A., Zhang, W., & Zverovitch, A. (2002). Experimental analysis of heuristics for the ATSP. In G. Gutin & A. Punnen (Eds.), *The Traveling Salesman Problem and Its Variations* (pp. 445–487). Norwell, MA, Kluwer Academic Publishers.

Johnson, D. S., & McGeoch, L. A. (1997). The travelling salesman problem: A case study in local optimization. In E. H. L. Aarts & J. K. Lenstra (Eds.), *Local Search in Combinatorial Optimization* (pp. 215–310). Chichester, UK, John Wiley & Sons.

Johnson, D. S., & McGeoch, L. A. (2002). Experimental analysis of heuristics for the STSP. In G. Gutin & A. Punnen (Eds.), *The Traveling Salesman Problem and Its Variations* (pp. 369–443). Norwell, MA, Kluwer Academic Publishers.

Johnson, D. S., Papadimitriou, C. H., & Yannakakis, M. (1988). How easy is local search? *Journal of Computer System Science*, *37*, 79–100.

Jünger, M., Reinelt, G., & Thienel, S. (1994). Provably good solutions for the traveling salesman problem. *Zeitschrift für Operations Research*, *40*, 183–217.

Kaelbling, L. P., Littman, M. L., & Moore, A. W. (1996). Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, *4*, 237–285.

Karaboga, D., & Pham, D. T. (2000). *Intelligent Optimisation Techniques*. Berlin, Springer-Verlag.

Khanna, A., & Zinky, J. (1989). The revised ARPANET routing metric. *ACM SIGCOMM Computer Communication Review*, *19*(4), 45–56.

Kirkpatrick, S., Gelatt, C. D., Jr., & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, *220*, 671–680.

Knox, J. (1994). Tabu search performance on the symmetric travelling salesman problem. *Computers & Operations Research*, *21*(8), 867–876.

Kolisch, R., & Hartmann, S. (1999). Heuristic algorithms for solving the resource constrained project scheduling: Classification and computational analysis. In J. Weglarz (Ed.), *Handbook on Recent Advances in Project Scheduling* (pp. 197–212). Dordrecht, Netherlands, Kluwer Academic Publishers.

Krasnogor, N., Hart, W. E., Smith, J., & Pelta, D. A. (1999). Protein structure prediction with evolutionary algorithms. In W. Banzhaf, J. Daida, A. E. Eiben, M. H. Garzon, V. Honavar, M. Jakiela, & R. E. Smith (Eds.), *Proceedings of the Genetic and Evolutionary Computation Conference*, vol. 2, (pp. 1596–1601). San Francisco, Morgan Kaufmann.

Krieger, M. J. B., & Billeter, J.-B. (2000). The call of duty: Self-organised task allocation in a population of up to twelve mobile robots. *Robotics and Autonomous Systems*, *30*(1–2), 65–84.

Krieger, M. J. B., Billeter, J.-B., & Keller, L. (2000). Ant-like task allocation and recruitment in cooperative robots. *Nature*, *406*, 992–995.

Krüger, F., Merkle, D., & Middendorf, M. (1998). Studies on a parallel Ant System for the BSP model. Unpublished manuscript.

Kube, C. R., & Zhang, H. (1994). Collective robotics: From social insects to robots. *Adaptive Behavior*, *2*, 189–218.

Kullback, S. (1959). *Information Theory and Statistics*. New York, John Wiley & Sons.

Kuntz, P., Snyers, D., & Layzell, P. (1999). A stochastic heuristic for visualizing graph clusters in a bi-dimensional space prior to partitioning. *Journal of Heuristics*, *5*(3), 327–351.

Laguna, M., & Martí, R. (2003). *Scatter Search: Methodology and Implementations in C*, vol. 24 of *Operations Research/Computer Science Interface*. Boston, Kluwer Academic Publishers.

Larrañaga, P., Kuijpers, C., Poza, M., & Murga, R. (1997). Decomposing Bayesian networks by genetic algorithms. *Statistics and Computing*, *7*(1), 19–34.

Larrañaga, P., & Lozano, J. A. (2001). *Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation. Genetic Algorithms and Evolutionary Computation*. Dordrecht, Netherlands, Kluwer Academic Publishers.

Lau, K. F., & Dill, K. A. (1989). A lattice statistical mechanics model of the conformation and sequence space of proteins. *Macromolecules*, *22*, 3986–3997.

Lawler, E. L. (1963). The quadratic assignment problem. *Management Science*, *9*, 586–599.

Lawler, E. L. (1976). *Combinatorial Optimization: Networks and Matroids*. New York, Holt, Rinehart, and Winston.

Lawler, E. L. (1977). A pseudopolynomial algorithm for sequencing jobs to minimize total tardiness. *Annals of Discrete Mathematics*, *1*, 331–342.

Lawler, E. L., Lenstra, J. K., Rinnooy Kan, A. H. G., & Shmoys, D. B. (1985). *The Travelling Salesman Problem*. Chichester, UK, John Wiley & Sons.

Lawrence, S., & Giles, C. L. (1998). Searching the world wide web. *Science*, *280*, 98–100.

Leguizamón, G., & Michalewicz, Z. (1999). A new version of Ant System for subset problems. In P. J. Angeline, Z. Michalewicz, M. Schoenauer, X. Yao, & A. Zalzala (Eds.), *Proceedings of the 1999 Congress on Evolutionary Computation (CEC'99)* (pp. 1459–1464). Piscataway, NJ, IEEE Press.

Leguizamón, G., & Michalewicz, Z. (2000). Ant Systems for subset problems. Unpublished manuscript.

Leguizamón, G., Michalewicz, Z., & Schütz, M. (2001). A ant system for the maximum independent set problem. In *Proceedings of the VII Argentinian Congress on Computer Science*, El Calafate, Santa Cruz, Argentina, vol. 2 (pp. 1027–1040).

Leighton, F. (1979). A graph coloring algorithm for large scheduling problems. *Journal of Research of the National Bureau of Standards*, *85*, 489–506.

Lenstra, J. K., Rinnooy Kan, A. H. G., & Brucker, P. (1977). Complexity of machine scheduling problems. In P. L. Hammer, E. L. Johnson, B. H. Korte, & G. L. Nemhauser (Eds.), *Studies in Integer Programming*, vol. 1 of *Annals of Discrete Mathematics* (pp. 343–362). Amsterdam, North-Holland.

Levine, J., & Ducatelle, F. (2003). Ant colony optimisation and local search for bin packing and cutting stock problems. *Journal of the Operational Research Society*, to appear.

Liang, Y.-C., & Smith, A. E. (1999). An Ant System approach to redundancy allocation. In P. J. Angeline, Z. Michalewicz, M. Schoenauer, X. Yao, & A. Zalzala (Eds.), *Proceedings of the 1999 Congress on Evolutionary Computation (CEC'99)* (pp. 1478–1484). Piscataway, NJ, IEEE Press.

Liaw, C.-F. (2000). A hybrid genetic algorithm for the open shop scheduling problem. *European Journal of Operational Research*, *124*(1), 28–42.

Lin, S. (1965). Computer solutions for the traveling salesman problem. *Bell Systems Technology Journal*, *44*, 2245–2269.

Lin, S., & Kernighan, B. W. (1973). An effective heuristic algorithm for the travelling salesman problem. *Operations Research*, *21*, 498–516.

Liu, J. S. (2001). *Monte Carlo Strategies in Scientific Computing*. New York, Springer-Verlag.

Lokketangen, A. (2000). Satisfied ants. In M. Dorigo, M. Middendorf, & T. Stützle (Eds.), *Abstract Proceedings of ANTS 2000—From Ant Colonies to Artificial Ants: Second International Workshop on Ant Algorithms* (pp. 73–77). Université Libre de Bruxelles, Brussels.

Lourenço, H., & Serra, D. (1998). Adaptive approach heuristics for the generalized assignment problem. Technical report No. 304, Universitat Pompeu Fabra, Department of Economics and Management, Barcelona, Spain.

Lourenço, H., & Serra, D. (2002). Adaptive search heuristics for the generalized assignment problem. *Mathware and Soft Computing*, *9*(2–3), 209–234.

Lourenço, H. R., Martin, O., & Stützle, T. (2002). Iterated local search. In F. Glover & G. Kochenberger (Eds.), *Handbook of Metaheuristics*, vol. 57 of *International Series in Operations Research & Management Science* (pp. 321–353). Norwell, MA, Kluwer Academic Publishers.

Lumer, E., & Faieta, B. (1994). Diversity and adaptation in populations of clustering ants. In J.-A. Meyer & S. W. Wilson (Eds.), *Proceedings of the Third International Conference on Simulation of Adaptive Behavior: From Animals to Animats* (pp. 501–508). Cambridge, MA, MIT Press.

Lundy, M., & Mees, A. (1986). Convergence of an annealing algorithm. *Mathematical Programming*, *34*, 111–124.

Maniezzo, V. (1999). Exact and approximate nondeterministic tree-search procedures for the quadratic assignment problem. *INFORMS Journal on Computing*, *11*(4), 358–369.

Maniezzo, V. (2000). Personal communication.

Maniezzo, V., & Carbonaro, A. (2000). An ANTS heuristic for the frequency assignment problem. *Future Generation Computer Systems*, *16*(8), 927–935.

Maniezzo, V., & Colorni, A. (1999). The Ant System applied to the quadratic assignment problem. *IEEE Transactions on Data and Knowledge Engineering*, *11*(5), 769–778.

Maniezzo, V., Colorni, A., & Dorigo, M. (1994). The Ant System applied to the quadratic assignment problem. Technical report IRIDIA/94-28, IRIDIA, Université Libre de Bruxelles, Brussels.

Maniezzo, V., & Milandri, M. (2002). An ant-based framework for very strongly constrained problems. In M. Dorigo, G. Di Caro, & M. Sampels (Eds.), *Proceedings of ANTS 2002—From Ant Colonies to Artificial Ants: Third International Workshop on Ant Algorithms*, vol. 2463 of *Lecture Notes in Computer Science* (pp. 222–227). Berlin, Springer-Verlag.

Marathe, M. V., Ravi, R., Ravi, S. S., Rosenkrantz, D. J., & Sundaram, R. (1996). Spanning trees short or small. *SIAM Journal on Discrete Mathematics*, *9*(2), 178–200.

Marchiori, E. (2002). Genetic, iterated, and multistart local search for the maximum clique problem. In S. Cagnoni, J. Gottlieb, E. Hart, M. Middendorf, & G. R. Raidl (Eds.), *Applications of Evolutionary Computing, Proceedings of EvoWorkshops 2002*, vol. 2279 of *Lecture Notes in Computer Science* (pp. 112–121). Berlin, Springer-Verlag.

Marchiori, E., & Steenbeek, A. (2000). An evolutionary algorithm for large scale set covering problems with application to airline crew scheduling. In *Real World Applications of Evolutionary Computing*, vol. 1083 of *Lecture Notes in Computer Science* (pp. 367–381). Berlin, Springer-Verlag.

Mariano, C. E., & Morales, E. (1999). MOAQ: An Ant-Q algorithm for multiple objective optimization problems. In W. Banzhaf, J. Daida, A. E. Eiben, M. H. Garzon, V. Honavar, M. Jakiela, & R. E. Smith (Eds.), *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-1999)*, vol. 1 (pp. 894–901). San Francisco, Morgan Kaufmann.

Martello, S., & Toth, P. (1990). *Knapsack Problems, Algorithms and Computer Implementations*. Chichester, John Wiley & Sons.

Martin, O., & Otto, S. W. (1996). Combining simulated annealing with local search heuristics. *Annals of Operations Research*, *63*, 57–75.

Martin, O., Otto, S. W., & Felten, E. W. (1991). Large-step Markov chains for the traveling salesman problem. *Complex Systems*, *5*(3), 299–326.

Martinoli, A., & Mondada, F. (1998). Probabilistic modelling of a bio-inspired collective experiment with real robots. In T. L. R. Dillman, P. Dario, & H. Wörn (Eds.), *Proceedings of the Fourth International Symposium on Distributed Autonomous Robotic Systems (DARS-98)* (pp. 289–308). Berlin, Springer-Verlag.

McQuillan, J. M., Richer, I., & Rosen, E. C. (1980). The new routing algorithm for the ARPANET. *IEEE Transactions on Communications*, *28*, 711–719.

Merkle, D., & Middendorf, M. (2000). An ant algorithm with a new pheromone evaluation rule for total tardiness problems. In S. Cagnoni, R. Poli, G. D. Smith, D. Corne, M. Oates, E. Hart, P. L. Lanzi, E. J. Willem, Y. Li, B. Paechter, & T. C. Fogarty (Eds.), *Real-World Applications of Evolutionary Computing*, vol. 1803 of *Lecture Notes in Computer Science* (pp. 287–296). Berlin, Springer-Verlag.

Merkle, D., & Middendorf, M. (2002a). Ant colony optimization with the relative pheromone evaluation method. In S. Gagnoni, J. Gottlieb, E. Hart, M. Middendorf, & G. Raidl (Eds.), *Applications of Evolutionary Computing: Proceedings of EvoWorkshops 2002*, vol. 2279 of *Lecture Notes in Computer Science* (pp. 325–333). Berlin, Springer-Verlag.

Merkle, D., & Middendorf, M. (2002b). Fast ant colony optimization on runtime reconfigurable processor arrays. *Genetic Programming and Evolvable Machines*, *3*(4), 345–361.

Merkle, D., & Middendorf, M. (2002c). Modeling the dynamics of ant colony optimization. *Evolutionary Computation*, *10*(3), 235–262.

Merkle, D., & Middendorf, M. (2003a). Ant colony optimization with global pheromone evaluation for scheduling a single machine. *Applied Intelligence*, *18*(1), 105–111.

Merkle, D., & Middendorf, M. (2003b). On the behavior of ACO algorithms: Studies on simple problems. In M. G. C. Resende & J. P. de Sousa (Eds.), *Metaheuristics: Computer Decision-Making, Combinatorial Optimization* (pp. 465–480). Boston, Kluwer Academic Publishers.

Merkle, D., Middendorf, M., & Schmeck, H. (2000a). Ant colony optimization for resource-constrained project scheduling. In D. Whitley, D. Goldberg, E. Cantu-Paz, L. Spector, I. Parmee, & H.-G. Beyer (Eds.), *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2000)* (pp. 893–900). San Francisco, Morgan Kaufmann.

Merkle, D., Middendorf, M., & Schmeck, H. (2000b). Pheromone evaluation in ant colony optimization. In *Proceedings of the 26th Annual Conference of the IEEE Electronics Society* (pp. 2726–2731). Piscataway, NJ, IEEE Press.

Merkle, D., Middendorf, M., & Schmeck, H. (2002). Ant colony optimization for resource-constrained project scheduling. *IEEE Transactions on Evolutionary Computation*, *6*(4), 333–346.

Merz, P., & Freisleben, B. (1997). Genetic local search for the TSP: New results. In T. Bäck, Z. Michalewicz, & X. Yao (Eds.), *Proceedings of the 1997 IEEE International Conference on Evolutionary Computation (ICEC'97)* (pp. 159–164). Piscataway, NJ, IEEE Press.

Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A., & Teller, E. (1953). Equation of state calculations by fast computing machines. *Journal of Chemical Physics*, *21*, 1087–1092.

Meuleau, N., & Dorigo, M. (2002). Ant colony optimization and stochastic gradient descent. *Artificial Life*, *8*(2), 103–121.

Michalewicz, Z. (1994). *Genetic Algorithms + Data Structures = Evolution Programs*. Berlin, Springer-Verlag.

Michalewicz, Z., & Fogel, D. B. (2000). *How to Solve It: Modern Heuristics*. Berlin, Springer-Verlag.

Michel, R., & Middendorf, M. (1998). An island model based Ant System with lookahead for the shortest supersequence problem. In A. E. Eiben, T. Bäck, M. Schoenauer, & H.-P. Schwefel (Eds.), *Proceedings of PPSN-V, Fifth International Conference on Parallel Problem Solving from Nature*, vol. 1498 of *Lecture Notes in Computer Science* (pp. 692–701). Berlin, Springer-Verlag.

Michel, R., & Middendorf, M. (1999). An ACO algorithm for the shortest supersequence problem. In D. Corne, M. Dorigo, & F. Glover (Eds.), *New Ideas in Optimization* (pp. 51–61). London, McGraw Hill.

Middendorf, M., Reischle, F., & Schmeck, H. (2002). Multi colony ant algorithms. *Journal of Heuristics*, *8*(3), 305–320.

Minton, S., Johnston, M., Philips, A., & Laird, P. (1992). Minimizing conflicts: A heuristic repair method for constraint satisfaction and scheduling problems. *Artificial Intelligence*, *58*(1–3), 161–205.

Mitchell, M. (1996). *An Introduction to Genetic Algorithms*. Cambridge, MA, MIT Press.

Mitchell, T. (1997). *Machine Learning*. Boston, McGraw Hill.

Mockus, J., Eddy, E., Mockus, A., Mockus, L., & Reklaitis, G. V. (1997). *Bayesian Heuristic Approach to Discrete and Global Optimization*. Dordrecht, The Netherlands, Kluwer Academic Publishers.

Moffett, M. W. (1988). Cooperative food transport by an Asiatic ant. *National Geographic Research*, *4*, 386–394.

Mondada, F., Franzi, E., & Ienne, P. (1993). Mobile robot miniaturization: A tool for investigation in control algorithms. In T. Yoshikawa & F. Miyazaki (Eds.), *Proceedings of the Third International Symposium on Simulation on Experimental Robotics (ISER-93)*, vol. 200 of *Lecture Notes in Control and Information Sciences* (pp. 501–513). Berlin, Springer-Verlag.

Morris, P. (1993). The breakout method for escaping from local minima. In *Proceedings of the 11th National Conference on Artificial Intelligence* (pp. 40–45). Menlo Park, CA, AAAI Press/MIT Press.

Morton, T. E., Rachamadugu, R. M., & Vepsalainen, A. (1984). Accurate myopic heuristics for tardiness scheduling. GSIA working paper 36-83-84, Carnegie Mellon University, Pittsburgh.

Moy, J. T. (1998). *OSPF Anatomy of an Internet Routing Protocol*. Boston, Addison-Wesley.

Mühlenbein, H. (1998). The equation for response to selection and its use for prediction. *Evolutionary Computation*, *5*(3), 303–346.

Mühlenbein, H., & Paass, G. (1996). From recombination of genes to the estimation of distributions. In W. Ebeling, I. Rechenberg, H.-P. Schwefel, & H.-M. Voigt (Eds.), *Proceedings of PPSN-IV, Fourth International Conference on Parallel Problem Solving from Nature*, vol. 1141 of *Lecture Notes in Computer Science* (pp. 178–187). Berlin, Springer-Verlag.

Navarro Varela, G., & Sinclair, M. C. (1999). Ant colony optimisation for virtual-wavelength-path routing and wavelength allocation. In P. J. Angeline, Z. Michalewicz, M. Schoenauer, X. Yao, & A. Zalzala (Eds.), *Proceedings of the 1999 Congress on Evolutionary Computation (CEC'99)* (pp. 1809–1816). Piscataway, NJ, IEEE Press.

Nawaz, M., Enscore, E., Jr., & Ham, I. (1983). A heuristic algorithm for the *m*-machine, *n*-job flow-shop sequencing problem. *OMEGA*, *11*(1), 91–95.

Nemhauser, G. L., & Wolsey, L. A. (1988). *Integer and Combinatorial Optimization*. Chichester, UK, John Wiley & Sons.

Nicolis, G., & Prigogine, I. (1977). *Self-Organisation in Non-Equilibrium Systems*. New York, John Wiley & Sons.

Nouyan, S. (2002). Agent-based approach to dynamic task allocation. In M. Dorigo, G. Di Caro, & M. Sampels (Eds.), *Proceedings of ANTS 2002—From Ant Colonies to Artificial Ants: Third International Workshop on Ant Algorithms*, vol. 2463 of *Lecture Notes in Computer Science* (pp. 28–39). Berlin, Springer-Verlag.

Nowicki, E., & Smutnicki, C. (1996a). A fast taboo search algorithm for the job-shop problem. *Management Science*, *42*(2), 797–813.

Nowicki, E., & Smutnicki, C. (1996b). A fast tabu search algorithm for the permutation flow-shop problem. *European Journal of Operational Research*, *91*(1), 160–175.

Nozaki, K., Ishibuchi, H., & Tanaka, H. (1997). A simple but powerful heuristic method for generating fuzzy rules from numerical data. *Fuzzy Sets and Systems*, *86*, 251–270.

Nyström, M. (1999). Solving certain large instances of the quadratic assignment problem: Steinberg's examples. Technical report, Department of Computer Science, California Institute of Technology, Pasadena.

Osman, I., & Laporte, G. (1996). Metaheuristics: A bibliography. *Annals of Operations Research*, *63*, 513–628.

Osman, I. H., & Kelly, J. P. (Eds.). (1996). *Meta-Heuristics: Theory and Applications*. Boston, Kluwer Academic Publishers.

Padberg, M. W., & Grötschel, M. (1985). Polyhedral computations. In E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan, & D. B. Shmoys (Eds.), *The Traveling Salesman Problem* (pp. 307–360). Chichester, UK, John Wiley & Sons.

Paessens, H. (1988). The savings algorithm for the vehicle routing problem. *European Journal of Operational Research*, *34*, 336–344.

Papadimitriou, C. H., & Steiglitz, K. (1982). *Combinatorial Optimization—Algorithms and Complexity*. Englewood Cliffs, NJ, Prentice Hall.

Papoulis, A. (1991). *Probability, Random Variables and Stochastic Process*, 3rd ed. New York, McGraw Hill.

Paquete, L., & Stützle, T. (2002). An experimental investigation of iterated local search for coloring graphs. In S. Cagnoni, J. Gottlieb, E. Hart, M. Middendorf, & G. R. Raidl (Eds.), *Applications of Evolutionary Computing, Proceedings of EvoWorkshops 2002*, vol. 2279 of *Lecture Notes in Computer Science* (pp. 122–131). Berlin, Springer-Verlag.

Parpinelli, R. S., Lopes, H. S., & Freitas, A. A. (2002a). An ant colony algorithm for classification rule discovery. In H. A. Abbass, R. A. Sarker, & C. S. Newton (Eds.), *Data Mining: A Heuristic Approach* (pp. 191–208). Hershey, PA, Idea Group Publishing.

Parpinelli, R. S., Lopes, H. S., & Freitas, A. A. (2002b). Data mining with an ant colony optimization algorithm. *IEEE Transactions on Evolutionary Computation*, *6*(4), 321–332.

Pearl, J. (1998). *Probabilisitic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Mateo, CA, Morgan Kaufmann.

Pelikan, M., Goldberg, D. E., & Lobo, F. (1999). A survey of optimization by building and using probabilistic models. Technical report IlliGAL, 99018, University of Illinois at Urbana-Champaign, Urbana, IL.

Pfahringer, B. (1996). Multi-agent search for open shop scheduling: Adapting the Ant-Q formalism. Technical report TR-96-09, Austrian Research Institute for Artificial Intelligence, Vienna.

Pimont, S., & Solnon, C. (2000). A generic ant algorithm for solving constraint satisfaction problems. In M. Dorigo, M. Middendorf, & T. Stützle (Eds.), *Abstract proceedings of ANTS 2000—From Ant Colonies to Artificial Ants: Second International Workshop on Ant Algorithms* (pp. 100–108). Université Libre de Bruxelles, Brussels.

Pinedo, M. (1995). *Scheduling—Theory, Algorithms, and Systems*. Englewood Cliffs, NJ, Prentice Hall.

Potts, C. N., & Wassenhove, L. N. V. (1991). Single machine tardiness sequencing heuristics. *IIE Transactions*, *23*, 346–354.

Quinlan, J. (1993a). *C4.5: Programs for Machine Learning*. San Francisco, Morgan Kaufmann.

Quinlan, J. (1993b). Combining instance-based and model-based learning. In *Proceedings of the Tenth International Conference on Machine Learning (ML-93)* (pp. 236–243). San Mateo, CA, Morgan Kaufmann.

Rajendran, C., & Ziegler, H. (2003). Ant-colony algorithms for permutation flowshop scheduling to minimize makespan/total flowtime of jobs. *European Journal of Operational Research*, to appear.

Rechenberg, I. (1973). *Evolutionsstrategie—Optimierung technischer Systeme nach Prinzipien der biologischen Information*. Freiburg, Germany, Fromman Verlag.

Reeves, C. (Ed.). (1995). *Modern Heuristic Techniques for Combinatorial Problems*. London, McGraw Hill.

Reimann, M., Doerner, K., & Hartl, R. F. (2002a). Insertion based ants for the vehicle routing problem with backhauls and time windows. In M. Dorigo, G. Di Caro, & M. Sampels (Eds.), *Proceedings of ANTS 2002—From Ant Colonies to Artificial Ants: Third International Workshop on Ant Algorithms*, vol. 2463 of *Lecture Notes in Computer Science* (pp. 135–148). Berlin, Springer-Verlag.

Reimann, M., Doerner, K., & Hartl, R. F. (2003). Analyzing a unified Ant System for the VRP and some of its variants. In G. R. Raidl, J.-A. Meyer, M. Middendorf, S. Cagnoni, J. J. R. Cardalda, D. W. Corne, J. Gottlieb, A. Guillot, E. Hart, C. G. Johnson, & E. Marchiori (Eds.), *Applications of Evolutionary*

Computing, Proceedings of EvoWorkshops 2003, vol. 2611 of *Lecture Notes in Computer Science* (pp. 300–310). Berlin, Springer-Verlag.

Reimann, M., Doerner, K., & Hartl, R. F. (2004). D-ants: Savings based ants divide and conquer the vehicle routing problem. *Computers & Operations Research*, *31*(4), 563–591.

Reimann, M., Stummer, M., & Doerner, K. (2002b). A savings based Ant System for the vehicle routing problem. In W. B. Langdon, E. Cantú-Paz, K. Mathias, R. Roy, D. Davis, R. Poli, K. Balakrishnan, V. Honavar, G. Rudolph, J. Wegener, L. Bull, M. A. Potter, A. C. Schultz, J. F. Miller, E. Burke, & N. Jonoska (Eds.), *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2002)* (pp. 1317–1325). San Francisco, Morgan Kaufmann.

Reinelt, G. (1991). TSPLIB—A traveling salesman problem library. *ORSA Journal on Computing*, *3*, 376–384.

Reinelt, G. (1994). *The Traveling Salesman: Computational Solutions for TSP Applications*, vol. 840 of *Lecture Notes in Computer Science*. Berlin, Springer-Verlag.

Resende, M. G. C., Pitsoulis, L. S., & Pardalos, P. M. (2000). Fortran subroutines for computing approximate solutions of weighted MAX-SAT problems using GRASP. *Discrete Applied Mathematics*, *100*(1–2), 95–113.

Resende, M. G. C., & Ribeiro, C. C. (2002). Greedy randomized adaptive search procedures. In F. Glover & G. Kochenberger (Eds.), *Handbook of Metaheuristics*, International Series in Operations Research & Management Science (pp. 219–249). Norwell, MA, Kluwer Academic Publishers.

Resnick, M. (1994). *Turtles, Termites, and Traffic Jams*. Cambridge, MA, MIT Press.

Robbins, H., & Monroe, H. (1951). A stochastic approximation method. *Annals of Mathematics and Statistics*, *22*, 400–407.

Robinson, G. E. (1992). Regulation of division of labor in insect societies. *Annual Review of Entomology*, *37*, 637–665.

Roli, A., Blum, C., & Dorigo, M. (2001). ACO for maximal constraint satisfaction problems. In *Proceedings of MIC'2001—Meta–heuristics International Conference*, vol. 1 (pp. 187–191). Porto, Portugal.

Romeo, F., & Sangiovanni-Vincentelli, A. (1991). A theoretical framework for simulated annealing. *Algorithmica*, *6*(3), 302–345.

Rossi, F., Petrie, C., & Dhar, V. (1990). On the equivalence of constraint satisfaction problems. In L. Carlucci Aiello (Ed.), *Proceedings of the 9th European Conference on Artificial Intelligence* (pp. 550–556). London, Pitman Publishing.

Roy, B., & Sussmann, B. (1964). Les problèmes d'ordonnancement avec constraintes disjonctives. Technical report DS No. 9bis, SEMA, Paris.

Rubin, P. A., & Ragatz, G. L. (1995). Scheduling in a sequence dependent setup environment with genetic search. *Computers & Operations Research*, *22*(1), 85–99.

Rubinstein, R. Y. (1981). *Simulation and the Monte Carlo Method.* New York, John Wiley & Sons.

Rubinstein, R. Y. (1999). The cross-entropy method for combinatorial and continuous optimization. *Methodology and Computing in Applied Probability*, *1*(2), 127–190.

Rubinstein, R. Y. (2001). Combinatorial optimization via the simulated cross-entropy method. In S. I. Gass & C. M. Harris (Eds.), *Encyclopedia of Operations Research and Management Science*. Boston, Kluwer Academic Publishers.

Sadeh, N., & Fox, M. (1996). Variable and value ordering heuristics for the job shop scheduling constraint satisfaction problem. *Artificial Intelligence*, *86*(1), 1–41.

Sadeh, N., Sycara, K., & Xiong, Y. (1995). Backtracking techniques for the job shop scheduling constraint satisfaction problem. *Artificial Intelligence*, *76*(1–2), 455–480.

Sahni, S., & Gonzalez, T. (1976). P-complete approximation problems. *Journal of the ACM*, *23*(3), 555–565.

Sait, S. M., & Youssef, H. (1999). *Iterative Computer Algorithms with Applications to Engineering*. Los Alamitos, CA, IEEE Computer Society Press.

Schoofs, L., & Naudts, B. (2000). Solving CSPs with ant colonies. In M. Dorigo, M. Middendorf, & T. Stützle (Eds.), *Abstract proceedings of ANTS 2000—From Ant Colonies to Artificial Ants: Second International Workshop on Ant Algorithms* (pp. 134–137). Université Libre de Bruxelles, Brussels.

Schoonderwoerd, R., Holland, O., & Bruten, J. (1997). Ant-like agents for load balancing in telecommunications networks. In *Proceedings of the First International Conference on Autonomous Agents* (pp. 209–216). New York, ACM Press.

Schoonderwoerd, R., Holland, O., Bruten, J., & Rothkrantz, L. (1996). Ant-based load balancing in telecommunications networks. *Adaptive Behavior*, *5*(2), 169–207.

Schreiber, G. R., & Martin, O. C. (1999). Cut size statistics of graph bisection heuristics. *SIAM Journal on Optimization*, *10*(1), 231–251.

Schrjiver, A. (2002). On the history of combinatorial optimization. Preprint available at www.cwi.nl/~lex/.

Schwefel, H.-P. (1981). *Numerical Optimization of Computer Models*. Chichester, UK, John Wiley & Sons.

Selman, B., & Kautz, H. (1993). Domain-independent extensions to GSAT: Solving large structured satisfiability problems. In *Proceedings of the 13th International Joint Conference on Artificial Intelligence* (pp. 290–295). San Francisco, Morgan Kaufmann.

Shang, Y., & Wah, B. W. (1998). A discrete Lagrangian-based global-search method for solving satisfiability problems. *Journal of Global Optimization*, *12*(1), 61–99.

Shankar, A. U., Alaettinoğlu, C., Dussa-Zieger, K., & Matta, I. (1992). Performance comparison of routing protocols under dynamic and static file transfer connections. *ACM Computer Communication Review*, *22*(5), 39–52.

Shmygelska, A., Aguirre-Hernández, R., & Hoos, H. H. (2002). An ant colony optimization algorithm for the 2D HP protein folding problem. In M. Dorigo, G. Di Caro, & M. Sampels (Eds.), *ANTS 2002*, vol. 2463 of *Lecture Notes in Computer Science* (pp. 40–52). Berlin, Springer-Verlag.

Shmygelska, A., & Hoos, H. H. (2003). An improved ant colony optimization algorithm for the 2D HP protein folding problem. In Y. Xiang, & B. Chaib-draa (Eds.), *Advances in Artificial Intelligence*, vol. 2671 of *Lecture Notes in Artificial Intelligence* (pp. 400–417). Berlin, Springer-Verlag.

Singh, S. P., & Sutton, R. S. (1996). Reinforcement learning with replacing eligibility traces. *Machine Learning*, *22*(1–3), 123–158.

Smith, B. M., & Dyer, M. E. (1996). Locating the phase transition in binary constraint satisfaction problems. *Artificial Intelligence*, *81*(1–2), 155–181.

Smith, D. H., Hurley, S., & Thiel, S. U. (1998). Improving heuristics for the frequency assignment problem. *European Journal of Operational Research*, *107*(1), 76–86.

Socha, K., Knowles, J., & Sampels, M. (2002). A $\mathcal{MAX}$-$\mathcal{MIN}$ Ant System for the university course timetabling problem. In M. Dorigo, G. Di Caro, & M. Sampels (Eds.), *Proceedings of ANTS 2002—From Ant Colonies to Artificial Ants: Third International Workshop on Ant Algorithms*, vol. 2463 of *Lecture Notes in Computer Science* (pp. 1–13). Berlin, Springer-Verlag.

Socha, K., Sampels, M., & Manfrin, M. (2003). Ant algorithms for the university course timetabling problem with regard to the state-of-the-art. In G. R. Raidl, J.-A. Meyer, M. Middendorf, S. Cagnoni, J. J. R. Cardalda, D. W. Corne, J. Gottlieb, A. Guillot, E. Hart, C. G. Johnson, & E. Marchiori (Eds.), *Applications of Evolutionary Computing, Proceedings of EvoWorkshops 2003*, vol. 2611 of *Lecture Notes in Computer Science* (pp. 334–345). Berlin, Springer-Verlag.

Solnon, C. (2000). Solving permutation constraint satisfaction problems with artificial ants. In W. Horn (Ed.), *Proceedings of the 14th European Conference on Artificial Intelligence* (pp. 118–122). Amsterdam, IOS Press.

Solnon, C. (2002). Ants can solve constraint satisfaction problems. *IEEE Transactions on Evolutionary Computation*, *6*(4), 347–357.

Steenstrup, M. E. (Ed.). (1995). *Routing in Communications Networks*. Englewood Cliffs, NJ, Prentice Hall.

Steinberg, L. (1961). The backboard wiring problem: A placement algorithm. *SIAM Review*, *3*, 37–50.

Sterling, T., Salmon, J., Becker, D. J., & Savarese, D. F. (1999). *How to Build a Beowulf*. Cambridge, MA, MIT Press.

Steuer, R. E. (1986). *Multiple Criteria Optimization: Theory, Computation and Application*. Wiley Series in Probability and Mathematical Statistics. New York, John Wiley & Sons.

Streltsov, S., & Vakili, P. (1996). Variance reduction algorithms for parallel replicated simulation of uniformized Markov chains. *Discrete Event Dynamic Systems: Theory and Applications*, *6*, 159–180.

Stützle, T. (1997a). An ant approach to the flow shop problem. Technical report AIDA-97-07, FG Intellektik, FB Informatik, TU Darmstadt, Germany.

Stützle, T. (1997b). $\mathcal{MAX}$-$\mathcal{MIN}$ Ant System for the quadratic assignment problem. Technical report AIDA-97-4, FG Intellektik, FB Informatik, TU Darmstadt, Germany.

Stützle, T. (1998a). An ant approach to the flow shop problem. In *Proceedings of the Sixth European Congress on Intelligent Techniques & Soft Computing (EUFIT'98)*, vol. 3 (pp. 1560–1564). Aachen, Germany, Verlag Mainz, Wissenschaftsverlag.

Stützle, T. (1998b). Parallelization strategies for ant colony optimization. In A. E. Eiben, T. Bäck, M. Schoenauer, & H.-P. Schwefel (Eds.), *Proceedings of PPSN-V, Fifth International Conference on Parallel Problem Solving from Nature*, vol. 1498 of *Lecture Notes in Computer Science* (pp. 722–731). Berlin, Springer-Verlag.

Stützle, T. (1999). *Local Search Algorithms for Combinatorial Problems: Analysis, Improvements, and New Applications*, vol. 220 of *DISKI*. Sankt Augustin, Germany, Infix.

Stützle, T., & Dorigo, M. (1999a). ACO algorithms for the quadratic assignment problem. In D. Corne, M. Dorigo, & F. Glover (Eds.), *New Ideas in Optimization* (pp. 33–50). London, McGraw Hill.

Stützle, T., & Dorigo, M. (1999b). ACO algorithms for the traveling salesman problem. In K. Miettinen, M. M. Mäkelä, P. Neittaanmäki, & J. Périaux (Eds.), *Evolutionary Algorithms in Engineering and Computer Science* (pp. 163–183). Chichester, UK, John Wiley & Sons.

Stützle, T., & Dorigo, M. (2002). A short convergence proof for a class of ACO algorithms. *IEEE Transactions on Evolutionary Computation*, *6*(4), 358–365.

Stützle, T., & Hoos, H. H. (1996). Improving the Ant System: A detailed report on the $\mathcal{MAX}$-$\mathcal{MIN}$ Ant System. Technical report AIDA-96-12, FG Intellektik, FB Informatik, TU Darmstadt, Germany.

Stützle, T., & Hoos, H. H. (1997). The $\mathcal{MAX}$-$\mathcal{MIN}$ Ant System and local search for the traveling salesman problem. In T. Bäck, Z. Michalewicz, & X. Yao (Eds.), *Proceedings of the 1997 IEEE International Conference on Evolutionary Computation (ICEC'97)* (pp. 309–314). Piscataway, NJ, IEEE Press.

Stützle, T., & Hoos, H. H. (1999). $\mathcal{MAX}$-$\mathcal{MIN}$ Ant System and local search for combinatorial optimization problems. In S. Voss, S. Martello, I. Osman, & C. Roucairol (Eds.), *Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization* (pp. 137–154). Dordrecht, Netherlands, Kluwer Academic Publishers.

Stützle, T., & Hoos, H. H. (2000). $\mathcal{MAX}$-$\mathcal{MIN}$ Ant System. *Future Generation Computer Systems*, *16*(8), 889–914.

Stützle, T., & Linke, S. (2002). Experiments with variants of ant algorithms. *Mathware and Soft Computing*, *9*(2–3), 193–207.

Subramanian, D., Druschel, P., & Chen, J. (1997). Ants and reinforcement learning: A case study in routing in dynamic networks. In *Proceedings of the 15th International Joint Conference on Artificial Intelligence* (pp. 832–838). San Francisco, Morgan Kaufmann.

Sudd, J. H. (1965). The transport of prey by ants. *Behaviour*, *25*, 234–271.

Sutton, R. S. (1988). Learning to predict by the methods of temporal differences. *Machine Learning*, *3*, 9–44.

Sutton, R. S., & Barto, A. G. (1998). *Reinforcement Learning: An Introduction*. Cambridge, MA, MIT Press.

Szwarc, W., Grosso, A., & Della Croce, F. (2001). Algorithmic paradoxes of the single machine total tardiness problem. *Journal of Scheduling*, *4*(2), 93–104.

Taillard, É. D. (1991). Robust taboo search for the quadratic assignment problem. *Parallel Computing*, *17*, 443–455.

Taillard, É. D. (1995). Comparison of iterative searches for the quadratic assignment problem. *Location Science*, *3*, 87–105.

Taillard, É. D. (1998). FANT: Fast Ant System. Technical report IDSIA-46-98, IDSIA, Lugano, Switzerland.

Taillard, É. D., Badeau, P., Gendreau, M., Guertin, F., & Potvin, J.-Y. (1997). A tabu search heuristic for the vehicle routing problem with soft time windows. *Transportation Science*, *31*, 170–186.

Tan, K. C., & Narashiman, R. (1997). Minimizing tardiness on a single processor with sequence-dependent setup times: A simulated annealing approach. *OMEGA*, *25*(6), 619–634.

Tanenbaum, A. (1996). *Computer Networks*. Englewood Cliffs, NJ, Prentice Hall.

Teich, T., Fischer, M., Vogel, A., & Fischer, J. (2001). A new ant colony algorithm for the job shop scheduling problem. In L. Spector, E. D. Goodman, A. Wu, W. B. Langdon, H.-M. Voigt, M. Gen, S. Sen, M. Dorigo, S. Pezeshk, M. H. Garzon, & E. Burke (Eds.), *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)* (pp. 803). San Francisco, Morgan Kaufmann.

Theraulaz, G., & Bonabeau, E. (1999). A brief history of stigmergy. *Artificial Life*, *5*, 97–116.

T'kindt, V., Monmarché, N., Tercinet, F., & Laügt, D. (2002). An ant colony optimization algorithm to solve a 2-machine bicriteria flowshop scheduling problem. *European Journal of Operational Research*, *142*(2), 250–257.

Toth, P., & Vigo, D. (Eds.). (2001). *The Vehicle Routing Problem*. SIAM Monographs on Discrete Mathematics and Applications. Philadelphia, Society for Industrial & Applied Mathematics.

van der Put, R. (1998). Routing in the faxfactory using mobile agents. Technical report R&D-SV-98-276, KPN Research, The Netherlands.

Vasquez, M., & Hao, J.-K. (2001). A hybrid approach for the 0-1 multidimensional knapsack problem. In *Proceedings of the 17th International Joint Conference on Artificial Intelligence* (pp. 328–333). San Francisco, Morgan Kaufmann.

Vazirani, V. V. (2001). *Approximation Algorithms*. Berlin, Springer-Verlag.

Voss, S., Martello, S., Osman, I. H., & Roucairol, C. (Eds.). (1999). *Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization*. Dordrecht, Netherlands, Kluwer Academic Publishers.

Voudouris, C. (1997). *Guided Local Search for Combinatorial Optimization Problems*. PhD thesis, Department of Computer Science, University of Essex, Colchester, UK.

Voudouris, C., & Tsang, E. (1995). Guided local search. Technical report CSM-247, Department of Computer Science, University of Essex, Colchester, UK.

Voudouris, C., & Tsang, E. P. K. (1999). Guided local search. *European Journal of Operational Research*, *113*(2), 469–499.

Wagner, I. A., Lindenbaum, M., & Bruckstein, A. M. (1996). Smell as a computational resource—A lesson we can learn from the ant. In M. Y. Vardi (Ed.), *Proceedings of the Fourth Israeli Symposium on Theory of Computing and Systems (ISTCS-99)* (pp. 219–230). Los Alamitos, CA, IEEE Computer Society Press.

Wagner, I. A., Lindenbaum, M., & Bruckstein, A. M. (1998). Efficient graph search by a smell-oriented vertex process. *Annals of Mathematics and Artificial Intelligence*, *24*, 211–223.

Wagner, I. A., Lindenbaum, M., & Bruckstein, A. M. (2000). ANTS: Agents, networks, trees and subgraphs. *Future Generation Computer Systems*, *16*(8), 915–926.

Wallace, R. J. (1996). Analysis of heuristic methods for partial constraint satisfaction problems. In E. Freuder (Ed.), *Principles and Practice of Constraint Programming—CP'96*, vol. 1118 of *Lecture Notes in Computer Science* (pp. 482–496). Berlin, Springer-Verlag.

Wallace, R. J., & Freuder, E. C. (1996). Heuristic methods for over-constrained constraint satisfaction problems. In M. Jampel, E. Freuder, & M. Maher (Eds.), *OCS'95: Workshop on Over-Constrained Systems at CP'95*, vol. 1106 of *Lecture Notes in Computer Science* (pp. 207–216). Berlin, Springer-Verlag.

Walrand, J., & Varaiya, P. (1996). *High-performance communication networks*. San Francisco, Morgan Kaufmann.

Walters, T. (1998). Repair and brood selection in the traveling salesman problem. In A. Eiben, T. Bäck, M. Schoenauer, & H.-P. Schwefel (Eds.), *Proceedings of PPSN-V, Fifth International Conference on Parallel Problem Solving from Nature*, vol. 1498 of *Lecture Notes in Computer Science* (pp. 813–822). Berlin, Springer-Verlag.

Wang, L. X., & Mendel, J. M. (1992). Generating fuzzy rules by learning from examples. *IEEE Transactions on Systems, Man, and Cybernetics*, *22*(6), 1414–1427.

Wang, Z., & Crowcroft, J. (1992). Analysis of shortest-path routing algorithms in a dynamic network environment. *ACM Computer Communication Review*, *22*(2), 63–71.

Wäscher, G., & Gau, T. (1996). Heuristics for the integer one-dimensional cutting stock problem: A computational study. *OR Spektrum*, *18*, 131–144.

Watkins, C. J., & Dayan, P. (1992). *Q*-Learning. *Machine Learning*, *8*, 279–292.

White, T., Pagurek, B., & Oppacher, F. (1998). Connection management using adaptive mobile agents. In H. R. Arabnia (Ed.), *Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'98)* (pp. 802–809). Las Vegas, NV, CSREA Press.

Whitley, D., Gordon, S., & Mathias, K. (1994). Lamarckian evolution, the Baldwin effect and function optimization. In Y. Davidor, H. Schwefel, & R. Männer (Eds.), *Proceedings of PPSN-III, Third International Conference on Parallel Problem Solving from Nature*, vol. 866 of *Lecture Notes in Computer Science* (pp. 6–15). Berlin, Springer-Verlag.

Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, *8*(3), 229–256.

Yagiura, M., Ibaraki, T., & Glover, F. (2004). An ejection chain approach for the generalized assignment problem. *INFORMS Journal on Computing*, to appear.

Zachariasen, M., & Dam, M. (1996). Tabu search on the geometric traveling salesman problem. In I. H. Osman & J. P. Kelly (Eds.), *Meta-heuristics: Theory and Applications* (pp. 571–587). Boston, Kluwer Academic Publishers.

Zlochin, M., Birattari, M., Meuleau, N., & Dorigo, M. (2001). Combinatorial optimization using model-based search. Technical report IRIDIA/2001-15, IRIDIA, Université Libre de Bruxelles, Brussels. To appear in *Annals of Operations Research*, 2004.

Zlochin, M., & Dorigo, M. (2002). Model-based search for combinatorial optimization: A comparative study. In J. J. Merelo, P. Adamidis, H.-G. Beyer, J.-L. Fernández-Villacanas, & H.-P. Schwefel (Eds.), *Proceedings of PPSN-VII, Seventh International Conference on Parallel Problem Solving from Nature*, vol. 2439 of *Lecture Notes in Computer Science* (pp. 651–661). Berlin, Springer-Verlag.

Index