ROBUST IMPLEMENTATION OF POLICIES USING ANT-LIKE AGENTS

Otto Wittner
Department of Telematics Norwegian University of Science and Technology
wittner@item.ntnu.no

Bjarne E. Helvik
Department of Telematics Norwegian University of Science and Technology
bjarne@item.ntnu.no

Abstract Policy based management is a powerful means for dealing with complex heterogeneous systems. However, the policies are commonly strictly interpreted, and it is tempting to resort to centralized decisions to resolve conflicts. At the same time, swarm intelligence based on "ant like" mobile agents has been shown to be able to deal with challenging optimization and trade-off problems. This paper discusses and demonstrates how policies may be used to govern the behavior of mobile agents to find near optimal solutions for the implementation of a set of potentially conflicting policies in a truly distributed manner. A more dependable/robust system is obtained. The enforcement of the policies is soft in the sense that it is probabilistic and yields a kind of "best effort" implementation. A case study illustrating how ant like mobile agents may implement load distribution and conflict free back-up policies, is presented.

Keywords: Policy enforcement, dependability, swarm intelligence, ant-like agents, mobile agents

1. Introduction

Today's computer and telecommunication network environments consists of a heterogeneous mix of network technologies, computing devices and applications. Policy based management [15] is a recognized concept for organizing management operations in such complex network environments. Results form research activities the last decade include standardization on policy management architectures, information models and protocols [11, 2] as well as a range of policy specification languages, cf. for instance Chapter 2.2 of [5] for a survey.

The traditional approach to policy enforcement yields basically an implementation of deterministic rules. Furthermore, in order to resolve potential conflicts between the different policies, there is a tendency to resort to meth-

The original version of this chapter was revised: The copyright line was incorrect. This has been corrected. The Erratum to this chapter is available at DOI: 10.1007/978-0-387-35620-4_43
D. Gaïti et al. (eds.), *Network Control and Engineering for QoS, Security and Mobility*

ods requiring centralized decision making [9]. Avoiding centralization is one reason (among others) for not always eliminating policy conflicts. Strategies for resolving policy conflicts have been studied by several researchers [9, 4].

In this paper we present a technology, known as *swarm intelligence* [1], applicable for enforcement of process policies in a distributed and robust manner. The fundamental components of swarm intelligence systems are simple, autonomous agents with a stochastic behavior similar to "agents" found in biological systems in nature, e.g. ant colonies. A high number of such agents are in action simultaneously communicating indirectly by changing the environment. The overall concept of swarm intelligence has several desirable properties from a dependability point of view: High redundancy, true distribution and adaptability. Section 2 of this paper introduces the fundamentals of swarm intelligence and agents with ant-like behavior. Such agents are proposed to implement policy management systems. The stochastic behavior of the agents is interpreted as *soft policy enforcement,* i.e. the policies specifying an agent's behavior will be enforced with some probability which yields a kind of "best effort" enforcement of conflicting policies and policies not fully enforceable due to resource limitations in the network. Section 3 describes how a swarm based system may implement a set policies, where subsection 3.3 describes how swarm based systems can resolve policy conflicts.

To demonstrate the potential of swarm intelligence implementations of policies, section 4 presents a case study where policies for connection management in a network are considered. Finally in section 5 we summarize, conclude and indicate future work.

2. Fundamental Agent Behavior

Swarm intelligence may be defined as "... *any attempt to design algorithms or distributed problem-solving devices inspired by collective behavior of social insect colonies and other animal societies*" [1]. Pioneering work in this field of research was performed decades ago, however the first attempts of applying swarm intelligence in a telecommunication management context was done by Steward and Appleby [16] and has later been developed by several other researchers [14, 3, 7].

2.1 Informal Behavior Outline

In this paper we concentrate on the Cross Entropy founded algorithm (CE algorithm) from [7] which is inspired by the foraging behavior of ants. The algorithm has a some fundamental elements in common with most swarm intelligence systems, with the extension that the search is successively focused by lowering a temperature parameter as better solutions are found. In addition, fundamental to our approach is the interaction between different types of ants as introduced in [19].

Many asynchronous, autonomous agents. A large number of autonomous simple agents move around in a given environment represented as a graph $\mathcal{G}$ with set $V(\mathcal{G})$ of vertices and $E(\mathcal{G})$ of edges. A unique type

of agent represents a policy combined with the target for enforcement of this policy. Enforcement is implemented by having agents utilize resources connected to vertices and edges, see section 4.2 for exemplification.

Stochastic search for solutions. Each agent search in a stochastic manner for a solution to a given problem, i.e. a policy enforcement, similar to ants' behavior when searching for a food source. A solution is represented as a path $\pi_t = \{si, ij, \ldots, kl, ld\}$ where $s, i, j, k, l, d \in V(\mathcal{G})$. An agent finds a path by doing a random walk combined with a search strategy (e.g. avoid revisiting vertices). Each step in the random walk is controlled by a probability vector $P_{t,i}$ where i is the currently visited vertex[1].

Indirect, asynchronous communication. Agents communicate indirectly and affect each others behavior by changing the environment. When a solution path π_t is found, an agent backtracks along the path and leaves virtual *pheromone* on every edge traversed. Real pheromones are chemical substances released by an ant to signal some message to other ants. Virtual pheromones are weights which influence the probability distributions $P_{t,i}$ which again influence the search behavior of the agents (cf. previous item).

Iterative with positive feedback. The search process is iterative and accelerated by positive feedback. When a solution is found and backtracking completed, an agent restarts forward search. Edges in high quality paths receive high pheromone weights. A high weight on an edge increase the probability for an agent to traverse the edge in successive searches. Pheromones placed as a result of successive searches reinforce the high weights even more. Over time high quality solutions emerge as sequences of highly weighted edges, resembling the emergence of ant trails. The CE algorithm speeds up this process further by making the agents more selective with respect to what a high quality solution is as better solutions are found [7, 12].

Together these components have been shown to enable design of systems capable of finding optimal or near optimal solution to complex optimization problems classified as NP-hard without any need of centralized control. Network management of large heterogeneous telecommunication networks involve solving different optimization problems with NP-hard complexity, e.g. routing and network planning. Decentralized techniques for performing such optimization are indeed of interest and may be worth incorporating into policy based network management systems.

[1]Each type of agent have their individual probabilities. For the sake of simplifying the introduction, the sub and superscripts necessary to identify the types are not shown.

2.2 Agent Algorithm

The actual steps in the CE algorithm that will be used in the case study of section 4, may be described as follows. Keep in mind that the algorithm describes the behavior of one type of agent. In multi-type agent systems, each type will have its separate set of parameters N, variables $P_{t,i}$, γ_t, functions L_{ij} and hence, $H(\pi_t, \gamma_t)$.

1 At iteration $t = 0$ initialize all probability vectors ($P_{t=0,i} \mid_{\forall i \in V(\mathcal{G})}$) to uniform distributions.

2 Create N agents. For each agent perform the step 3-5.

3 Generate a path π_t by performing a random walk based on the probability vectors $P_{t,i} \mid_{\forall i \in V(\mathcal{G})}$ and some search strategy ([18]).

4 At the destination vertex calculate, the optimal Boltzmann temperature γ_t which controls $H(\pi_t, \gamma_t) = e^{-\frac{L(\pi_t)}{\gamma_t}}$, the performance function returning the quality of path π_t. $L(\pi_t) = \sum_{ij \in \pi_t} L_{ij}$ is the raw cost of path π_k which again is the sum of edge costs for all edges in the path. The edge cost is related to the success of the policy enforcement on the edges as will be discussed in section 4. The optimal solution for γ_t will result in a certain amplification of high quality paths and a minimum weighted average of all path qualities for paths found so far, i.e. search focus is directed towards good candidate solution paths. The enforcement of potentially conflicting policies are "signaled" through L_{ij} which may include calculations based on pheromones from alien species of ant-like agents, see section 3.3.

5 Using γ_t and $H(\pi_t, \gamma_t)$ from step 4, generate new probability vectors $P_{t+\delta,i}$ by minimizing the cross entropy between $P_{t,i} \mid_{\forall i}$ and $P_{t+\delta,i} \mid_{\forall i}$, where δ is the incremental time to the next agent starts its search. Minimizing the cross entropy ensures an optimal shift in the probabilities with respect to γ_t and the performance function. The generation of $P_{t+\delta,i} \mid_{\forall i}$ is achieved by adjusting relevant probabilities of $P_{t,i} \mid_{\forall i}$ during re-traversal of path π_t.

6 Repeat steps 3-5 until $H(\widehat{\pi}, \gamma_t) \approx H(\widehat{\pi}, \gamma_{t+\Delta})$ where $\widehat{\pi}$ is the best path found and Δ is an appropriately chosen time.

For details on the CE algorithm and the formal foundations the reader is referred to [12, 7]

3. From Policies to Agent Behavior

The agent behavior, described in the previous section, has hereto been adapted and used to handle specific optimization problems, either as stand alone "optimization engines", e.g. [12, 8, 6], or as an integrated part of network O&M, [7, 19, 18]. However we suggest to take a policy view point of the

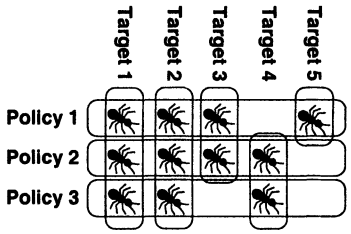


Figure 1. Policies and target organization. Each policy target instance is allocated one unique type of ant-like agent.

problem at hand and base the agent behavior on a set of policies. Soft enforcement of policies may be regarded as minimizing the deviation from the policies under constraints set by resource limitations in the network, network topology and operation. In the following subsections we describe the steps from policy specification to policy implementation by ant-like agents. We create policies at two levels, and in accordance with [10] we apply the term *functional policies* for upper level policies and *process policies* for lower level policies.

3.1 Problem Fundamentals: Functional Policies

Elements connecting the agent behavior to a specific problem are the *graph* ($\mathcal{G}$) and the *edge costs* (L_{ij}).

The *graph* requires a mapping onto the solution space for the problem at hand. A one-to-one mapping is desirable such that all possible solutions from the solution space are represented in the graph and no solutions being none-existing in the solution space can be found in the graph. In section 4.2 nodes and link resources in an IP backbone network are mapped to graph vertices and edges respectively.

Given a graph representation, the edge costs and the path performance function control the core behavior of an agent. As the first step in the design process for defining these elements, we suggest to *establish a set of policies,* $\{Fr\}_{r=0,\ldots,R}$, *which identify requirements for relevant targets fundamental in the management problem.* It is assumed that these policies are ordered according to their importance, hence, r is referred to as the rank of the policy.

Targets of the management problem are the managed entities in the network which shall comply with one or more of the policies. The sub- or superscript m refers to a target. Examples of targets are virtual paths, cf. the case study, and databases. In general, targets are associated with vertices, edges as well as patterns of vertices and/or edges in the graph. Figure 1 illustrates how policies and targets may be organized. Some targets may be treated by several policies while others only by one or a few. In section 4.1 two functional policies, F1 and F2, are established each treating 12 different target paths which again are sequences of link resources.

As a step in the enforcement implementation of the policies we allocate one unique type of ant-like agent to each policy-target instance, mr, as illustrated

in Figure 1. Thus a type of ant-like agent is intended to be designed specifically to enforce a certain policy for a certain target.

3.2 Behavior Details: Process Policies

To better understand what behavior details will be required for our agents, we refine the functional policies from the previous section and establish a set of process policies, i.e., each functional policy is mapped to a set of process policies, $Fr \rightarrow \{Pr1, \ldots, Prn_r\}$.

The next step is to establish a measure of fulfillment of the requirements stated by the process polices. The measure will indicate how desirable the execution of a specific action is and will typically be related to vertices and/or edges in the graph. For the sake of simplicity, we regard edges only. The fulfillment with respect to policy-target instance mr on edge ij is a function given by the process policies, i.e., $L_{ij}^{mr} = f_{P1,\ldots,Pn}(\{a_{ij}\}, \{c_{ij}\})$ where $\{a_{ij}\}$ is the set of requirements/demands associated with the edge and $\{c_{ij}\}$ is the set of resources available on the edge to meet these requirements.

3.3 Resolving Policy Conflicts by Detestation

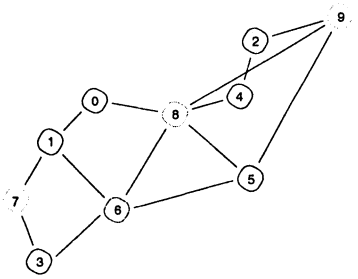
During enforcement of several policy-target instances (Figure 1) conflicts may arise. If limitations in the resources required by several policy target instances exists, one or more policy target instances may not be enforced. Hence some sort of conflict resolution mechanism is required.

An ant-like agent allocated to a policy target instance uses a unique pheromone type, i.e. the agent maintains a reserved set of variables on each vertex and/or edge in the graph which indicate if the specific part of the graph is likely to be part of the final solution enabling enforcement of the policy-target instance. Agents update only their own pheromones but may still read alien agents' pheromones.

As a mechanism for conflict resolution we include pheromone levels of alien agents as weights in the cost function. If alien pheromones indicate that a required resource is likely to be allocated by one or more alien agents, the weights will increase the cost of accessing the resource and eventually result in a reduced probability for the agent in question to revisit the resource in the next search for a policy enforcement solution. This effect can be seen as having *agents with conflicting interests detest each other* [17, 19]. In section 4.2 agents detest each other with different degrees depending on what policy-target instance they are allocated to. Semi-formally, we extend our edge cost function to $L_{ij}^{mr} = g_{P1,\ldots,Pn}(\{a_{ij}\}, \{c_{ij}\}, \{P_{t,i}^{ns}\})$ where $\{P_{t,i}^{ns}\}$ is the pheromone trail left by all other types of agents, and it is implicit that the policies may cause conflicts.

3.4 Soft Policy Enforcement

Two aspects of the agent behavior result in a less than 100% guaranty of finding enforcement solution for all policy target instances.



Connection	Primary Path	Backup Path
1–6	1–6	1–7–3–6
1–8	1–0–8	1–6–8
1–9	1–0–8–9	1–7–3–6–5–9
6–1	6–1	6–3–7–1
6–8	6–8	6–5–8
6–9	6–5–9	6–8–4–2–9
8–1	8–0–1	8–6–1
8–6	8–6	8–5–6
8–9	8–9	8–4–2–9
9–1	9–8–0–1	9–5–6–3–7–1
9–6	9–5–6	9–2–4–8–6
9–8	9–8	9–2–4–8

Figure 2. Left: The Norwegian university IP backbone network consisting of 10 nodes and 14 duplex links (i.e. 28 simplex links). Right: A pattern of paths for an optimal policy enforcement.

Firstly, the fundamental stochastic behavior of the agents combined the size of the problem solution space renders it impossible to guaranty all potential solutions evaluated. However in [12] clear indications are given that CE algorithms with high probability converge towards a near optimal solution.

Secondly, in complex networks with multiple conflicting policies, a solution complying with all policies may not exist. The detestation weights introduced to handle policy conflicts will, however, force agents to search in parts of the solution space where the policies are partially enforced. This is likely to result in solutions enforcing most policies to such a degree that the overall system behavior is satisfactory, i.e., a best effort policy enforcement.

The stochastic behavior of the agents as well as the detestation properties adds a potential ability of adaptation to the environment [13]. Should the target network environment change and require changes in the policy specification, the agents may still be able to find satisfactory sets of policy enforcement solutions. We look at this effect as *soft policy enforcement* which we believe may provide better control over complex network environments than traditional static policy enforcement implementations.

4. Case Study: Primary and Backup Path Reservation

In this section we perform a policy-to-agent-behavior case study of a primary and backup path reservation scenario. The scenario presented is constructed but still realistic and may in the near future become highly relevant.

Figure 2 shows a graph illustrating the Norwegian university IP backbone infrastructure. All links have 155 Mb/s capacity. Node 6, 1, 8 and 9 are the university cities Oslo, Bergen, Trondheim and Tromsø respectively. A team of physicians from the faculties of medicine at the four different universities have decided to establish a virtual research environment based on high quality multimedia conferencing over the IP network. Such multimedia streams requires up to 70 Mb/s capacity thus 70 Mb/s connections from each university to all others are required, i.e. 12 simplex connections with both primary and

backup paths. A backup path is used only if the corresponding primary path has a failure.

4.1 Establishing Policies

The following functional policies are specified with the intention of enabling the desired transport service. Policy targets are emphasized:

Policy F1 Reserve required bandwidth capacity in the network for *primary paths for all specified connections* such that no links are overloaded (zero traffic loss).

Policy F2 Reserve required bandwidth capacity in the network for *backup paths for all specified connections* such that when a single link failure occurs, traffic loss due to link overload is minimized .

The functional policies are further refined resulting in a set of process policies with the mappings $F1 \rightarrow \{Pr1\}$ and $F2 \rightarrow \{Pr2, Pr3, Pr4\}$. In the policy descriptions below *contend for capacity* should be interpreted as causing a link overload if all parties sharing a link transmit simultaneously. However, sharing a link does not necessary imply contention for capacity given the link in question has enough total capacity. Further, π_r^m represent a path of rank r for connection m. Rank is either primary (0) or backup (1), i.e. $r \in \{0, 1\}$.

Policy Pr1 Reserve link capacity for connection m's primary path π_0^m such that π_0^m *do not contend for capacity against other primary paths* $\pi_0^n \mid_{\forall n}$ *on any of the links in* π_0^m (i.e. each link in π_0^m must have enough capacity to carry connection m in addition to all other primary paths carried).

Policy Pr2 Reserve link capacity for connection m's backup path π_1^m such that π_1^m *do not contend for capacity against primary paths* $\pi_0^n \mid_{\forall n}$ *on any of the links in* π_1^m (i.e. each link in π_1^m must have enough capacity to carry connection m in addition to all other primary paths carried).

Policy Pr3 Reserve link capacity for connection m's backup path π_1^m such that all links in π_1^m are *disjoint from* links in connection m's *primary path* π_0^m (i.e. no single link failure causes both a primary and backup path to fail).

Policy Pr4 Reserve link capacity for connection m's backup path π_1^m such that π_1^m *do not contend for capacity against connection* n*'s backup path* π_1^n when connection m and n's primary paths π_0^m and π_0^n are not disjoint (i.e. each link shared by π_1^m and π_1^n must have enough capacity to carry both connection m and n should a link shared by π_0^m and π_0^n fail).

Policy Pr1 ensures no overload (zero traffic loss) when primary paths are in use. Policy Pr2, Pr3 and Pr4 ensures minimum traffic loss due to overload when a single link failure occurs. In Figure 2 a pattern of paths is given which enforces all policies, i.e. an optimal policy implementation.

4.2 Implementing Policies

To implement soft policy enforcement the network environment is mapped onto a graph structure such that vertices represent network nodes and edges represent link resources. Unique types of ant-like agents are allocated to the policy-target instances. Two classes of ant-like agents are created, one class to search for primary paths, and one for backup paths. Primary agents/paths are of rank 0, and backup agents/paths of rank 1. Each of the 12 connections are allocated one species. Every species has both primary and backup agents, thus a total of 24 different types of ant-like agents are implemented each with a unique pheromone type and a specific policy target for which to find an enforcement solution.

Taken from policy F1 and F2, link overload is chosen as the cost measure for a link. Policies Pr1-Pr4 require link capacity to be reserved. Due to limited resources, reservations initiated by two policies may result in an overloaded link, i.e. a policy conflict is experienced. As an attempt to resolve such conflicts agents are made to detest pheromones related to alien paths. Combining the cost measure and pheromone detestation the following link cost expression is derived:

$$L_{ij}^{mr} = \mathcal{S}\left[a_m + \sum_{\forall ns:\, ij \in \pi_s^n} P_{ij}^{ns}\, V_i^{ns}\, Q_{mr}^{ns}\, a_n - c_{ij}\right]$$

where L_{ij}^{mr} is the expected potential link overload on link ij if the link is included in connection m's path of rank r. The function, terms and factors of the expression can be explained and related to policies as follows:

- To avoid negative cost values and smoothen the transmission between no loss and loss, a shaping function

$$\mathcal{S}[c] = \begin{cases} \eta \cdot e^{\frac{c}{e \cdot \eta}}, & c < \eta \cdot e \\ c, & otherwise \end{cases}$$

  is applied to the link cost expression. η is a parameter ($\mathcal{S}[0] = \eta$).

- a_m, a_n and c_{ij} represent capacities required by connection m and n, and capacity available on link ij respectively. The sum of all required capacities less available capacity $[a_m + \sum_n a_n - c_{ij}]^+$ equals link overload, thus implements the quality measure specified in policies F1 and F2.

- P_{ij}^{ns} and V_i^{ns} are approximate probabilities. P_{ij}^{ns} indicates the probability that agent of rank s for connection n (agent ns) will include link ij in its solution (i.e. follow ij during search) given that it visits node i. V_i^{ns} indicates the probability that agent ns will include node i in its solution, i.e. visit i during search. Hence P_{ij}^{ns} and V_i^{ns} represent pheromones for agent ns and weight the alien capacity request a_n according to how likely it is to be used when the search process converges. P_{ij}^{ns} and V_i^{ns} implement detestation and enforcement of policies Pr1-Pr3.

- Q_{mr}^{ns} is a weight function controlling the level of detestation

$$Q_{mr}^{ns} = \begin{cases} \widehat{Q}_{m,r-1}^{n,s-1}, & n \neq m,\ s = r = 1 & \text{– the likelihood that path } ns \text{ and } mr \text{ do not have disjont primary paths, i.e. enforcement of Pr4} \\ 1, & n \neq m,\ s = r = 0 & \text{– i.e. no influence by } Q_{mr}^{ns} \\ 20, & n = m,\ s < r = 1 & \text{– strong detestation and enforcement of Pr3} \\ 5, & n \neq m,\ s < r = 1 & \text{– medium detestation and enforcement of Pr2} \\ 0, & otherwise & \end{cases}$$

Finally the cost $L(\pi_r^m)$ of a path π_r^m is made the sum of the costs of all links in the path

$$L(\pi_r^m) = \sum_{ij \in \pi_r^m} L_{ij}^{mr}$$

The sum of all link cost is an approximation of the desirable path cost measure indicated by policies F1 and F2. Optimally only the cost of the link in the path responsible for the largest loss should represent the path cost. However such maximization violates the requirement of additivity in the cost function for CE algorithms (chap. 5 [12]). The sum of link cost is a conservative approximation thus no infeasible solutions are included in the search space by this.

4.3 Simulation results

Table 1 shows simulation results for the scenario. All values are averaged over 12 simulations and standard deviations are given in brackets. The first column indicates which policy the results in the last three columns are related to. The second column gives a short description of the policy. The third column shows the average number of incidents causing policy enforcement failures. The fourth column shows expected relative permanent loss (in percent of total traffic load) due to failed policy enforcements, and the last column shows expected relative loss due to failed policy enforcements given a single link failure. The last row of Table 1 show totals, both overall expected relative loss including and excluding permanent loss.

In total 12 primary and 12 backup paths are established in each simulation. Solutions for enforcement of policy Pr1 are found in 11 out of 12 simulations, i.e. only one incident over all simulations is observed where 3 primary paths share a link and traffic loss of 55 Mb/s is experienced. In the case of policy Pr2, 3 incidents of enforcement failures are on average observed, and given a single link failure the expected relative loss due to such incidents is on average 1.179 % . For the two last process policies, Pr3 and Pr4, 0.9 incidents of failed enforcement is observed on average, and a relative loss of around 0.2 % is expected given a single link failure.

Table 1. Results averaged over 12 simulations.

Policy	Description	No of incidents of failed enforcement	Expected relative loss (%)	
			Permanent	On single link failure
Pr1	Primary v.s. primary path	0.08 (±0.29)	0.504 (±1.816)	0.504 (±1.816)
Pr2	Primary v.s. alien backup path	3.0 (±1.15)	0	1.179 (±0.773)
Pr3	Primary v.s. own backup path	0.9 (±1.25)	0	0.229 (±0.347)
Pr4	Backup v.s. none-disjoint primary paths	0.9 (±1.14)	0	0.162 (±0.241)

Totals, given a single link failure:	Incl. permanent loss	Excl. permanent loss
Expected relative loss (%)	2.074 (±1.560)	1.570 (±0.770)

Summarized, the results from Table 1 give encouraging indications that ant-like agents can produce policy enforcement implementations of high quality. In our example, implementations are produced enabling close to full enforcement of all policies. Given a single link failure an overall loss of 2% of total traffic is expected. However there is still room for improvement especially considering that at least one optimal enforcement implementation exist (no loss given a single link failure) as given in Figure 2.

The poorest results are experienced for policy Pr2. We suspect one reason could be the lack of a additional policy which specifies what strategy the allocation process for primary paths should have when encountering well established alien backup paths, i.e. a policy similar to Pr2 but with opposite roles for primary and backup path. Introducing such a policy would imply having primary agents detest backup agents as well as opposite, which again should result in overall less contention for capacity.

5. Conclusion

In this paper we have presented a distributed management approach which is based on cooperating (and competing) simple mobile agents forming a swarm intelligent policy management system. We have described a design process where a policy specification is transformed into an ant-like agent optimization system, capable of finding enforcement solution for the policies. The agent system's ability to resolve policy conflicts is termed *soft policy enforcement*. Results from a simulation scenario indicate that near optimal enforcement solutions can by found by the agent system. However no guaranty for finding optimal solutions can be given.

Ongoing work include large scale pheromone management. When embedding swarm based policy enforcement in a large network environment, care must be taken to avoid overloading nodes with pheromone data. Future work should include formalizing the design process as well as further testing of the soft policy enforcement scheme in dynamic network environments.

References

[1] E. Bonabeau, M. Dorigo, and G. Theraulaz. *Swarm Intelligence: From Natural to Artifical Systems*. Oxford University Press, 1999.

[2] J. Boyle, R. Cohen, D. Durham, S. Herzog, R. Rajan, and A. Sastry. RFC2748: The COPS (Common Open Policy Service) Protocol. IEFT, January 2000.

[3] G. D. Caro and M. Dorigo. AntNet: Distributed Stigmergetic Control for Communications Networks. *Journal of Artificial Intelligence Research*, 9:317–365, Dec 1998.

[4] J. Chomicki, J. Lobo, and S. Naqvi. A logic programming approach to conflict resolution in policy management. In *KR2000: Principles of Knowledge Representation and Reasoning*, pages 121–132, San Francisco, 2000. Morgan Kaufmann.

[5] N. C. Damianou. *A Policy Framework for Management of Distributed Systems*. PhD thesis, Imperial College of Science, Technology and Medicine, University of London, Departement of Computing, February 2002.

[6] M. Dorigo. Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem. *IEEE Transactions on Evolutionary Computing*, 1(1), April 1997.

[7] B. E. Helvik and O. Wittner. Using the Cross Entropy Method to Guide/Govern Mobile Agent's Path Finding in Networks. In *Proceedings of 3rd International Workshop on Mobile Agents for Telecommunication Applications*. Springer Verlag, August 14-16 2001.

[8] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by Simulated Annealing. *Science 220*, pages 671–680, 1983.

[9] E. Lupu and M. Sloman. Conflicts in Policy-based Distributed Systems Management. *IEEE Transactions on Software Engineering - Special Issue on Inconsistency Management*, 25(6):852–869, Nov. 1999.

[10] M. J. Maullo and S. B. Calo. Policy Management: An Architecture and Approach. In *Proceedings of the IEEE First International Workshop on Systems Management, 1993*, pages 13 –26, UCLA, California, April 1993.

[11] B. Moore, E. Ellesson, J. Strassner, and A. Westerinen. RFC3060: Policy Core Information Model - Version 1 Specification. IETF, February 2001.

[12] R. Y. Rubinstein. The Cross-Entropy Method for Combinatorial and Continuous Optimization. *Methodology and Computing in Applied Probability*, pages 127–190, 1999.

[13] R. Y. Rubinstein. *Stochastic Optimization: Algorithms and Applications*, chapter Combinatorial Optimization, Cross-Entropy, Ants and Rare Events - Section 7: Noisy Networks. Kluwer Academic Publishers, 2001.

[14] R. Schoonderwoerd, O. Holland, J. Bruten, and L. Rothkrantz. Ant-based Load Balancing in Telecommunications Networks. *Adaptive Behavior*, 5(2):169–207, 1997.

[15] M. S. Sloman. Policy Driven Management for Distributed Systems. *Journal of Network and Systems Management*, 2(4):333–360, 1994.

[16] S. Steward and S. Appleby. Mobile Software Agents for Control of Distributed Systems Based on Principles of Social Insect Behavior. *BT Technology Journal*, 12(2):104–113, April 1994.

[17] G. N. Varela and M. C. Sinclair. Ant Colony Optimisation for Virtual-Wavelength-Path Routing and Wavelength Allocation. In *Proceedings of the Congress on Evolutionary Computation (CEC'99)*, Washington DC, USA, July 1999.

[18] O. Wittner and B. E. Helvik. Cross-Entropy Guided Ant-like Agents Finding Cyclic Paths in Scarcely Meshed Networks. In *The Third International Workshop on Ant Algorithms, ANTS'2002*, Brussels, Belgium, Sept 2002.

[19] O. Wittner and B. E. Helvik. Cross Entropy Guided Ant-like Agents Finding Dependable Primary/Backup Path Patterns in Networks. In *Proceedings of Congress on Evolutionary Computation (CEC2002)*, Honolulu, Hawaii, May 12-17th 2002. IEEE.