Computational Social Dilemmas

Natalie S. Glance†, *Tad Hogg*‡

†Rank Xerox Research Centre, Grenoble Laboratory.
6 chemin de Maupertuis, 38240 Meylan, France.
Email: glance@xerox.fr

‡Xerox Palo Alto Research Center.
3333 Coyote Hill Rd., Palo Alto, CA 94304.
Email: hogg@parc.xerox.com

May 23, 1995

Abstract

World-wide interlinked computer networks are forming the foundation for computational societies of software agents. Already, these new societies have encountered problems endemic to human communities, such as over-using common resources with thrashing over virtual memory and competition by packets for network time. Unlike with human societies, these inefficiencies can be overcome by re-working the algorithms governing the protocols. However, the public good problem, in which a common good is available to all regardless of contribution, can arise computationally in more subtle ways. We discuss how this can happen using Braess' Paradox and demonstrate that adding resources to a computational system can counterintuitively lower the overall performance. Thus, this is a case in which distributed algorithms are provably unable to achieve globally optimal performance. We illustrate our claim using a genetic algorithm and computational ecosystem. We further argue that inefficiencies arising from public good problems will occur commonly in computational societies of agents that choose among strategies based on utilities that depend upon the choices of other agents. Finally, we suggest borrowing several constructs from human societies to avoid common good problems in computational ones, such as taxation, privatization, and the establishment of norms.

Key words: Braess' Paradox, distributed systems, common goods.

1 Introduction

We are in the midst of constructing vast societies of computational agents [1, 2, 3, 4, 5]. These agents often act autonomously using locally available information. Often, such decentralized systems behave more robustly than traditional ones based on a single central controller, because the latter respond more rapidly to changes and perform better. They are also simpler to design and incrementally modify. On the other hand, agents, like people in human societies, face the difficult task of performing well in spite of incomplete, imperfect, and changing information about their environment. What lessons might human societies offer designers of computational ones?

For example, the provision of public goods causes difficulties for autonomous agents [6]. These goods are available to all, regardless of whether an individual contributes ("cooperates") or not ("defects"). Common examples are national defense and cleaning up the environment. Specifically, the dilemma arises when

$$B > c > b \tag{1}$$

where B is the benefit to the entire group of an additional individual's contribution, c is the cost to the individual of cooperating, and b is the individual benefit received from cooperating. Agents in this situation are tempted to defect since their cost for cooperating exceeds the benefit they receive. When everyone reasons similarly no public good is produced, and everyone is worse off because the overall social benefit of cooperation exceeds the combined costs. Thus, individually rational behavior leads to an overall suboptimal result.

The nature of the computational society determines whether such dilemmas can be avoided. At one extreme, the agents perform tasks for a variety of users with differing goals, including communicating over the network, sharing available workstations for distributed processing, and searching a database. In this case, the dilemma can be avoided by converting public goods to private ones via simple computational pricing and accounting mechanisms [7] coupled with adequate security to prevent cheating by actively malicious programs such as computer viruses.

At the other extreme are societies of agents all designed with a single goal in mind, e.g., solving a problem via parallel processing. Naturally, all the agents benefit by producing the desired final solution. While this sharing of benefits creates an opportunity for a social dilemma, one might expect that it could be avoided by simply programming in virtues that are lauded but under-represented in human history: altruism, cooperation, and rationality. Thus, public goods seem an excellent example of a problem which should not exist in societies of computational agents that are designed to address a single problem.

However, the public good problem is as much a computational dilemma as a social one, and occurs in two senses. The weak sense of the dilemma underlines the general mathematical fact that locally optimal actions often result in suboptimal global performance; at the same time, determining the globally superior choice by central fiat may be computationally intractable. In the strong sense, the overall minimum cost (or maximum performance) may actually be provably *impossible* to find within a distributed setting since individual choices will take the system away from the minimum instead of towards it: this is the computational analogue of a social dilemma. Computational social dilemmas thus stand in contrast to other hard problems whose hilly solution landscapes cause methods such as neural nets [8], simulated annealing [9] and greedy heuristics for NP-hard search problems [10] to become trapped in local minima, making the overall minimum hard, but *not* impossible, to find.

This discussion raises the following issues [11]: in the absence of central control, what forms do public goods problems take, how can they be recognized, and how might they be resolved. In this paper we consider these questions in the context of Braess' Paradox, one manifestation of the public good problem that we describe in the next section, which reveals how social dilemmas can arise in purely computational domains. The consequences are then illustrated with a genetic algorithm [12, 13] and a computational ecosystem [1]. We then examine the significance of these results by showing that dilemmas from the broader category of public goods can arise in a wide range of circumstances. We also present some methods that can eliminate these dilemmas or reduce their consequences when they do arise.

2 Braess' Paradox

Social dilemmas, as defined by Eq. 1, cause inefficiencies because the equilibrium state does not yield optimal performance. A particularly subtle version of the social dilemma is Braess' paradox [14, 15] in which adding resources to a system causes lower performance by moving the equilibrium away from the optimal configuration. Fig. 1 shows a simple example. Agents enter the network from the bottom and choose among various paths, moving between nodes along the indicated links. The traffic is described by the fraction of agents taking each link, and the cost for moving on a link can depend on that link's flow. Intuition suggests that adding a new link between nodes **B** and **C** would always improve performance. However, the following argument shows this is not always the case.

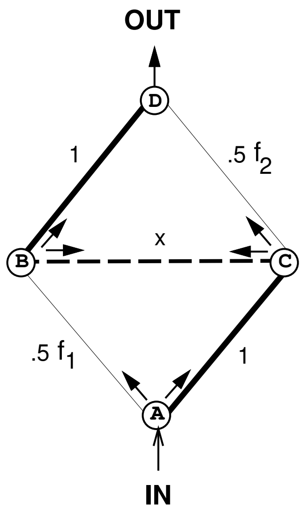

Figure 1: An example of Braess' paradox. Alternative choices and their costs (e.g., traffic delay) for flow through the network from bottom to top are indicated. Throughput is lower with the horizontal link than without it when agents choose the path that minimizes their individual cost. Decision points are marked by the circles.

At equilibrium the flow through the network is distributed so that each agent incurs the same cost and so that any change by one agent would increase its cost, given that none of the other agents change their routes. The overall equilibrium is reached by local decisions in which each agent minimizes its own cost given the behavior of the other agents. When the link between nodes **B** and **C** is absent, half the agents choose the left-hand path and half choose the right-hand path. This results in optimal flow: the average cost per agent is minimized and equals 1.25.

When the horizontal link with cost $1/4 < x < 1/2$ is present, an agent traveling from node **A** to **D** now has four paths to choose from (instead of only two). The most expensive of these is path **ACBD** which costs $c_{ACBD} = 2 + x$ and is consequently taken only in error. Path **ABD** costs the agent $c_{ABD} = 1 + f_1/2$ (where f_1 is the fraction of agents following arc **AB**), path **ACD** costs $c_{ACD} = 1 + f_2/2$ (f_2 is the fraction following arc **CD**) and path **ABCD** costs $c_{ABCD} = f_1/2 + x + f_2/2$. The overall global average cost is then given by the sum of costs on each path weighted by the fraction of agents following that path:

$$\begin{aligned} C_{global} &= (f_1 - f_x)c_{ABD} + (1 - f_1)c_{ACD} + f_x c_{ABCD} \\ &= \left(\frac{3}{2} - f_1 + f_1^2\right) + \frac{f_x^2}{2} + f_x(x - f_1) \end{aligned} \tag{2}$$

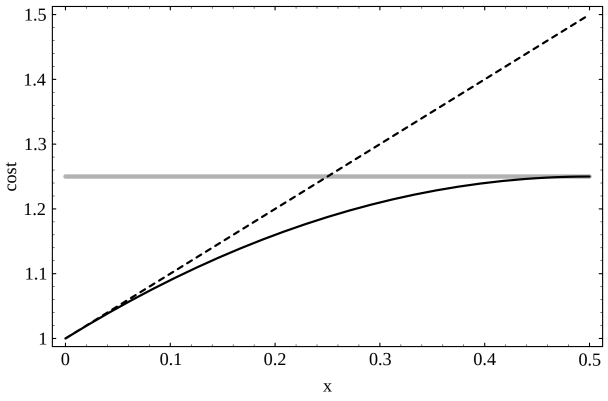

Figure 2: Comparison of costs for the network as a function of the cost of the cross link, x. The solid curve is the global minimum of the average cost per agent. The dashed curve is the cost obtained when each agent makes locally optimal decisions. For comparison, the gray line is the cost when there is no cross link.

Given perfect information, an agent will always prefer path **ABCD** to the other two since $f_1/2+x<1$ and $f_2/2+x<1$, independent of the congestion levels f_1 and f_2. If each agent minimizes its own costs, then each takes path **ABCD**, giving $f_1=f_x=1$ and an average cost of $1+x$ which is greater than the average cost of 1.25 without the link. Thus, when agents individually minimize cost, additional resources counterintuitively lead to lower overall performance, as illustrated in Fig. 2. The global minimum of Eq. 2 occurs at $f_1=1-x,\ f_x=1-2x$ giving

$$C_{min}=1+x-x^2. \tag{3}$$

The addition of the extra link in the network effectively creates a social dilemma for the agents, whose individual choices impact the global costs. To make the dilemma clear, we will map Braess' Paradox onto the family of social dilemmas represented by Eq. 1. First, we construct an overall utility function from the average transit cost per agent: $U_{total}=-n\,C_{global}$, where n is the number of agents flowing through the system. Then we associate the change in transit cost due to an agent's path choice with the cost for cooperation and the change in average transit cost per agent with the increase or decrease in utility. We will show that the system may benefit overall when the agent picks a path with higher transit cost. The change in total utility will then reflect the sum of the global benefit and the individual's cost due to cooperation:

$$\Delta U_{total}=B-c. \tag{4}$$

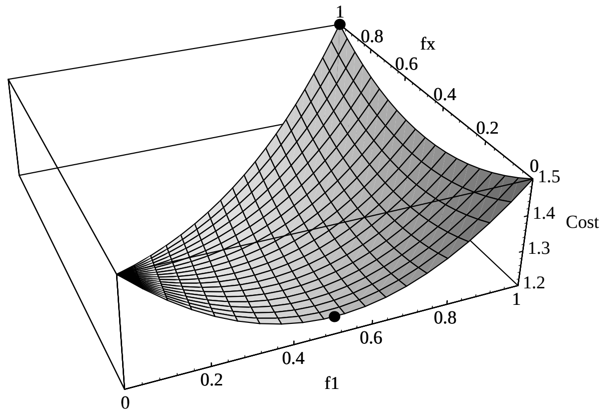

Figure 3: Average cost per agent as a function of the fraction f_1 of agents that follow arc **AB** and of the fraction $f_x \leq f_1$ that follow the new link connecting nodes **B** and **C**, when its cost is $x = 0.5$. Without the link, half the agents follow the left-hand path from **A** to **D** and half follow the right-hand path. This distribution of agents minimizes the average cost per agent, with or without the extra link. When the link is added, the system shifts to the new equilibrium in which all agents follow path **ABCD** through a series of individually rational choices. However, the new equilibrium at $f_1 = f_x = 1$ is clearly suboptimal, demonstrating the potentially deleterious effect of distributed decision-making in this class of situations.

Consider the case $f_x > f_1 - x$, where f_x is the fraction of agents traveling along arc **BC** (an analogous construction exists for $f_x < f_1 - x$). If we associate cooperation with the individual action that optimizes global performance and defection with the action that optimizes local performance at the expense of the global good, then, in this case, an agent at node **B** cooperates by following arc **BD** and defects by following arc **BC**. The cost c to the agent for cooperating is the difference between the costs of paths **ABD** and **ABCD**, using $f_2 = 1 - f_1 + f_x$:

$$c = \frac{f_1}{2} - \frac{f_x}{2} + \frac{1}{2} - x. \tag{5}$$

Since $f_x \leq f_1$ and $x < 1/2$, the cost of cooperation is always positive, making cooperation a sacrificial move for the agent. On the other hand, if the agent does cooperate (incurring cost c), then the average transit cost per agent goes down. The change in average transit cost is given roughly by $(dC_{global}/df_x)\Delta f_x$, ignoring terms of order $1/n$. Since f_x decreases by $1/n$ when an agent cooperates, $\Delta f_x = -1/n$. Putting it all together, the overall utility increases by

$$\Delta U_{total} \simeq -n(-\frac{1}{n})\frac{dC_{global}}{df_x} = -f_1 + f_x + x. \tag{6}$$

From Eq. 4, the global benefit B is then given by

$$B \equiv \Delta U_{total} + c = -\frac{f_1}{2} + \frac{f_x}{2} + \frac{1}{2}. \tag{7}$$

In this case, the individual benefit b of cooperation is zero. Since $f_x > f_1 - x$, we get $B > c > b$, so the new link creates a social dilemma (the public goods problem, though not Braess' Paradox, remains even for $0 < x < \frac{1}{4}$).

The difference in average cost per agent due to individual actions can also be understood using Fig. 3 for $x = 0.5$. Individual defections move the system along increasing f_x, which reduces an individual's cost but always increases total cost when $f_x > f_1 - x$. If agents individually optimize their paths the system moves away from the global minimum $f_1 = f_2 = \frac{1}{2}$, $f_x = 0$ to the clearly inferior state $f_1 = f_x = 1$ with each agent choosing to follow the new link between nodes **B** and **C**. Thus, the distributed decision-making of the agents moving through the network by optimizing individual preferences takes the system far from the optimal overall state which an omniscient central controller would readily locate.

3 Computational Examples

3.1 Genetic Algorithms

To see how this paradox can affect a computational society, we turn to genetic algorithms (GAs) [12, 13]. This is a search method based on analogy with biological evolution. GA's breed collections of structures (often represented by bit strings) to find those that minimize a given cost function, or, equivalently, maximize a fitness function. Specifically, we use a GA to solve the complete network flow problem of Fig. 4 in which Braess' Paradox (Fig. 1) is embedded. A publicly available genetic algorithm implementation [16] was used to conduct the experiments. In each generation exactly one agent is breeded, corresponding to asynchronous choices by the agents.[1]

In our example, the structures manipulated by the GA encode the paths taken by the agents: one structure per agent. By construction, the GA

[1] This asynchronous reproduction is necessary to model continuous time reproduction. Synchronous reproduction, in which all of the agents reproduce at once, models either discrete time or synchronized systems [17]. Thus the entire population of ten agents reproduces once every ten generations.

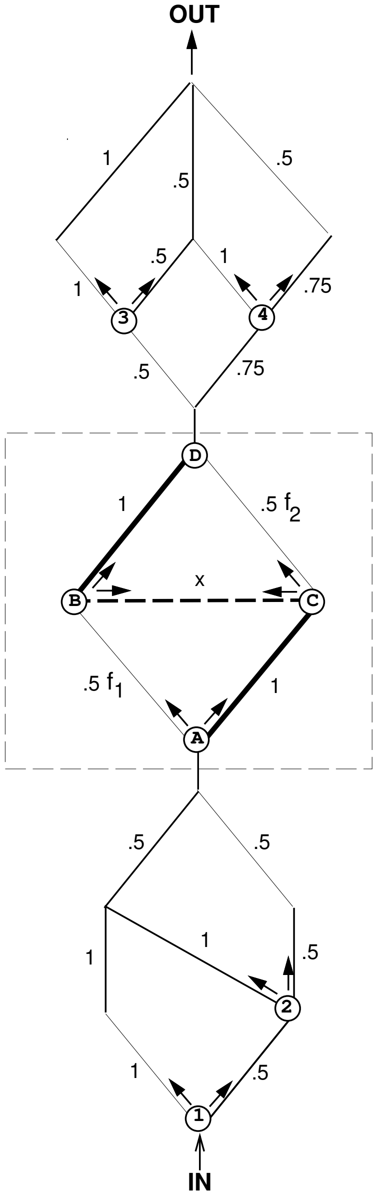

Figure 4: A larger network embedding Braess' paradox. Alternative choices and their costs (e.g., traffic delay) for flow through the network from bottom to top are indicated. The boxed region incorporates the paradox, giving lower throughput with the horizontal link than without it when agents choose the path that minimizes their individual cost. Decision points are marked by the circles.

evolves the collection of structures in a direction that tends to decrease the individual transit cost for the agents. Since Braess' Paradox is embedded within the network, we hypothesize that including the extra link **BC** will result in decreased global performance when the agents make local decisions.

Typical results are shown in Fig. 5: with the additional link the GA is unable to find the optimal state. To show that this was not exceptional, we repeated this experiment 100 times running the GA 2000 generations each time, and averaged the cost over all the agents for generations 1500 to 2000. The extra link always lowered the performance of the GA even though the cheapest path through the network costs 4.25 per agent with or without the extra link.

Excluding two experiments which had not converged to their final configuration by generation 1500, the mean cost for the case without the extra link (no dilemma) was 4.275 with standard deviation 0.0076. Adding the link increased the average cost to 4.51 with standard deviation 0.0056. (The average costs are higher than predicted by the cost function of Fig. 2 — 4.25 without the link, 4.5 with the link — because mutations introduce fluctuations which always increase cost once the population has converged.) Thus, the additional link on average increased the average cost by 0.235, almost the full 0.25 predicted by Braess' paradox.[2] The difference is somewhat smaller than expected because the mutation-caused fluctuations induce more variation in the no dilemma case, since the average fitness is then very sensitive to the distribution of agents among the left and ride-hand paths within the boxed-in region of the network.

Note that, if we had not been aware of the dilemma's role in this example, we might have thought the lowered performance of the GA was due to random variations in the population of agents. We might then have been tempted to let the GA continue evolving the population, hoping for eventual improvement. Alternatively, we might have concluded that the decreased performance resulted from a local minimum and instead restarted the GA from a new set of initial conditions. For our situation, neither of these common reactions would in fact be beneficial. Social dilemmas thus offer an alternative perspective on why problems may be difficult for GAs to solve besides the well-known problems of random fluctuations and local minima [18].

The results would be very different if the GA bred structures that solved the global problem of finding a set of structures to minimize the total cost

[2]Note that this difference is statistically significant since the standard deviations in the estimates of the means are only one tenth the size of the standard deviations given above, when combined via the square root of the sum of squares to get the standard deviation of the difference, we get 0.00094, much smaller than difference between 0.235 and 0.25.

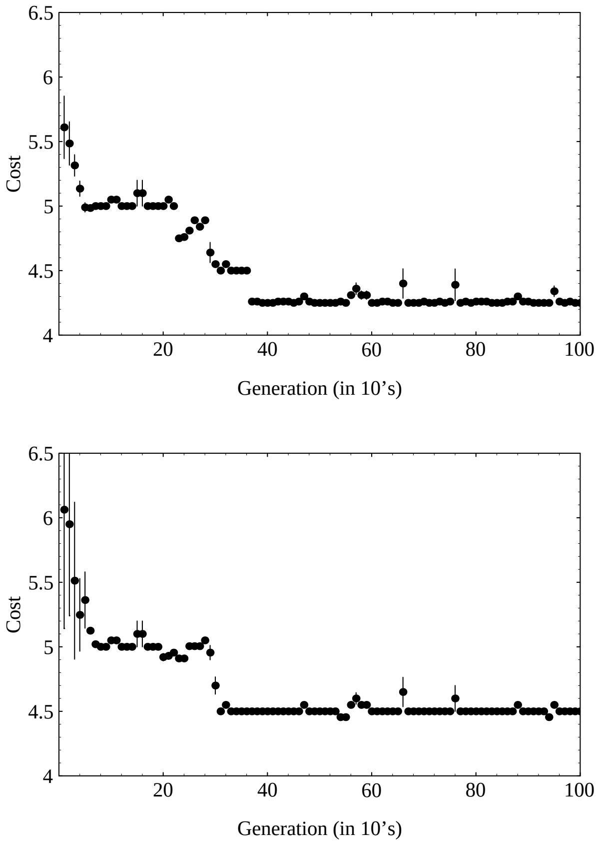

Figure 5: Behavior of genetic algorithm for a case without dilemma (top) and with dilemma (bottom). The cost of the extra link in the network, present in the dilemma case, was set to $x = 0.5$. With or without the dilemma, the cheapest path through the network costs 4.25, but this solution is found only when the extra link is not present. The plots show the transit costs through the network as a function of generation number (in tens). The points give the average cost while the bars indicate the range of costs. We used a population of size 10, with crossover rate 0.5, mutation rate 0.01, and a generation gap of 0.1 (i.e., exactly one agent breeded per generation). Six bits were used to represent the path taken by each agent through the network.

to the whole group of agents. However, the GA would then be solving the problem of finding the optimal strategy for a central control mechanism. Furthermore, such an approach is possible only if the overall cost function is known, which will not generally be the case in a distributed setting.

3.2 Computational Ecosystems

A second illustration of the effect of Braess' Paradox is given by computational ecosystems [1]. These consist of agents with computational tasks to perform and various resources with which to accomplish their tasks. The benefits associated with the resources are computational measures of performance, such as the time to complete a task or the accuracy of the solution. The suitability of a resource can depend on how many other agents are using it. For instance, additional load on a machine can reduce the execution speed of each task on that machine. Alternatively, agents using a resource could assist one another in their computations. For example, if the overall task could be decomposed into many communicating subtasks, the agents may be better off using the same computer rather than running more rapidly on separate machines and then being limited by slow communications. When the system is large and the information on the suitability of various resources is imperfect, delayed, and changing over time, it is not feasible to use a central agent to allocate resources. Instead, flexibility and responsiveness to local changes require that autonomous agents asynchronously select among available resources. A simple model of these societies assumes that each agent uses one resource at a time, evaluates its choice at a rate α and selects that resource it perceives to be best. These choices lead to a range of dynamical behaviors including simple equilibria, continual oscillations and chaos.

To see the consequence of Braess' Paradox, we can use the costs of the various paths of Fig. 1 as the costs of resources in a computational ecosystem, i.e., use resources 1, 2 and 3 whose costs correspond to paths **ABD**, **ACD** and **ABCD** respectively. Let F_i be the fraction of agents using resource i and C_i the respective cost. These are related to the fractions using the various links in Fig. 1 by $F_1 = f_1 - f_x$, $F_2 = 1 - f_1$ and $F_3 = f_x$, which are also the corresponding fractions for each path as used in Eq. 2. Since $F_1 + F_2 + F_3 = 1$ we can express the costs using only F_1 and F_2 as

$$\begin{aligned} C_1 &= \frac{1}{2}(3 - F_2) \\ C_2 &= \frac{1}{2}(3 - F_1) \\ C_3 &= 1 + x - \frac{1}{2}(F_1 + F_2) \end{aligned} \tag{8}$$

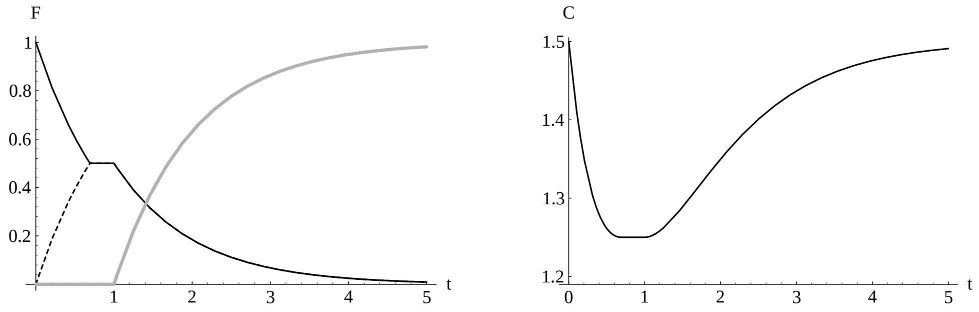

Figure 6: Behavior as a function of time for a computational ecosystem facing resource costs that include Braess' Paradox. We use $F_1(0) = 1$ and $\alpha = 1$. Initially there are two resources, but at $t = 1$ the third resource, with $x = 0.5$, is added. The left figure shows the fraction of agents using resources 1, 2 and 3 (solid, dashed and gray, respectively). On the right is the global average cost C for the system. Note that the addition of the third resource causes the cost to rise from its minimum value of 1.25 up to 1.5.

and the overall average cost for the system is given by $C = \sum F_i C_i$. Let ρ_i be the probability that resource i is perceived to be the best one when an agent makes a choice. Then $\sum \rho_i = 1$ and the ρ_i depend on the F_i because through the resource costs. The dynamical behavior is then given by (derived in [1])

$$\frac{dF_i}{dt} = -\alpha(F_i - \rho_i). \tag{9}$$

The simplest situation is when the agents have full information on their individual costs and there are no delays in the information. In that case they will always select the resource with the lowest cost, giving $\rho_i = 1$ when C_i is the lowest cost, and zero otherwise. When there are only two resources (or equivalently x is so large that resource 3 is never selected), $\rho_1 = 1$ when $C_1 < C_2$, (which implies $F_1 < F_2$) and similarly $\rho_2 = 1$ when $F_1 > F_2$. Thus, starting from an initial condition of all agents using resource 1 ($F_1 = 1$), the resource loads smoothly change as $F_1(t) = e^{-\alpha t}$ and $F_2(t) = 1 - F_1(t)$ until their values reach 1/2, at which point they remain at that value. When the third resource is added, the agents will flock to it because of its lower cost ($\rho_3 = 1$). Thus, $F_1$and F_2 will decay exponentially toward zero, while F_3 increases towards one. This behavior is illustrated in Fig. 6, along with its consequence for the overall performance.

The dilemmas presented here are only one of several causes for inefficiencies. Previous studies of the 3–resource case have shown that lowered performance can be also caused by oscillations and chaotic behaviors due to uncertainty and delays [19]. Since the existence of a dilemma will not always be readily apparent from the cost functions [20], especially in more complex cases that involve nonlinearities, it may be easy to confuse the lowered performance due to a social dilemma with that due to uncertainty or delays in the available information.

4 How Often Will Dilemmas Arise?

So far we've presented two examples of computational societies in which a social dilemma–like situation can arise, further illustrating Braess' Paradox. We now turn to the question of how often social dilemmas from the broader category of public good problems can be expected to occur. Ideally, we'd like to be able to approach this question empirically; we'd like to be able to examine real systems of coordinated computational agents to test for the occurrence of dilemmas. For example, we could generate thousands of random samples of cooperating agents solving a graph problem and test each case for the presence or not of a dilemma. While this approach would yield very pertinent results, it presents many obstacles, such as how to define realistic classes of random problems. Instead we propose to gain insight into the frequency of occurrence through a more theoretical approach.

If we take another look at the two examples of computational societies we've presented, the commonalities are striking, despite the superficial differences. Let's start off with a common vocabulary: agents, strategies, utility functions, and localized forces. In the case of genetic algorithms, as mentioned earlier, the structures bred by the algorithm can be thought of as agents. These structures can also be thought of as phenotypes, in analogy with genetics, and the semantics they encode, the genotype, or more generally, the strategy taken by the agent. The fitness function also maps directly onto utility: the higher an agent's fitness, then the higher its utility for adopting the particular strategy. Note we're using the word strategy loosely here, since it is chosen by selection forces, not by the agent. That is, we use it in the same sense biologists might talk about the mating strategies of horseshoe crabs (and humans?), encoded rather than volitional [21]. The localized force, then, in this case, is selection in proportion to fitness (or utility), along with crossover and mutation operators on the strategies. It is localized because it depends on individual fitness rather than average group fitness.

The mapping onto the common vocabulary is also very straightforward in the case of computational ecosystems. We already have agents; the choice of resource is really just a very simple strategy, with a one-to-one mapping between resources and strategies. The utility of a particular strategy becomes the negative of the analogous resource cost (or some monotonic function thereof). Because of this relationship between utility and costs, we will speak of maximizing utility, whereas before we spoke of minimizing costs. The localized mechanism for change that drives the time evolution of the system in this case is, at least superficially, volitional. Agents compare the utilities of the various strategies to decide which one to follow.

We also need a general way of talking about how the systems evolve. For computation ecosystems we have an explicit approximation to the dynamics, given by Eq. 9. In the case of perfect information, the agents in a computational ecosystem will evolve towards an equilibrium point in which no agent can improve its utility by switching strategies. Although it is not possible to make such a strong statement regarding genetic agents bred by selection because mutation and crossover will always cause fluctuations away from equilibrium, we can in general identify almost frozen configurations of agents with the equilibrium point. However, since the dynamics of Eq. 9 only represents average behavior of a computational ecosystem, its true dynamics will exhibit fluctuations away from the equilibrium point as well [19].

We expect that many other types of computational societies (some of which we discuss in Section 6) can be described in the common vocabulary of agents, strategies, utilities, and localized forces. The dilemma trap will appear whenever the local dynamics does not lead the agents into the globally optimal state even when the agents have perfect local information. Below, we look at linear utility functions to understand how commonly the optimal state does not map onto the equilibrium, hence giving rise to a public goods problem, although one perhaps not as severe as that found with Braess' paradox. We then generalize the linear theory to handle non-linear utility functions. We will find that non-dilemma cases are the exception, rather than the norm, although the effect of dilemmas in bumpy utility terrains will compete with the presence of sub-optimal equilibrium points to degrade the performance of the society.

4.1 Theory for Linear Costs

First, the state of a computational society, outlined in general terms above, can be characterized by the fraction of agents f_i adopting strategy i with

$$\sum_{i=1}^{s} f_i = 1, \tag{10}$$

where s is the number of available strategies. Consider the situations with s strategies and with utilities that depend linearly on their frequency (i.e., the fraction of agents following the strategy):

$$\mathbf{U} = M\,\mathbf{f} + \mathbf{v}. \tag{11}$$

The vectors $\mathbf{U}$ and $\mathbf{f}$ have components U_i and f_i that give the the utility and frequency of the strategies for agent i. The matrix M specifies how the utilities depend on the frequency distribution of the strategies and the vector $\mathbf{v}$ is a frequency-independent contribution to the utilities. Note that because of Eq. 10, the choices for M and $\mathbf{v}$ are not unique; this relation can be used to eliminate f_s from the costs so that the last column of M will be zero. Furthermore, because agents make choices based on the maximum utility option (when they have perfect information), the results presented here will not be affected by rescaling the unit of utility, i.e., we can multiply each component of M and $\mathbf{v}$ by a constant or shift the origin of the utilities by adding a constant to each component of $\mathbf{v}$.

The global average utility is given by

$$U_{global} = \sum f_i\,U_i = \mathbf{f}^T M\,\mathbf{f} + \mathbf{f}\cdot\mathbf{v}. \tag{12}$$

From this we see that the linear utility example considered here is particularly useful for investigating social dilemmas: except in degenerate special cases, C_{global} has only one minimum and so avoids the complication of local minima. The strategy distribution that maximizes this utility is found by maximizing $C_{global} - \lambda(\sum f_i - 1)$, where the Lagrange multiplier λ is used to enforce Eq. 10. Setting all partial derivatives to zero gives the extremum as the solution to

$$A_G \begin{pmatrix} \mathbf{f} \\ -\lambda \end{pmatrix} = \begin{pmatrix} \mathbf{v} \\ 1 \end{pmatrix} \tag{13}$$

with

$$A_G = \begin{pmatrix} M^T + M & \mathbf{1} \\ \mathbf{1} & 0 \end{pmatrix} \tag{14}$$

where $\mathbf{1}$ represents the vector all of whose components equal 1. There are two degenerate cases in which this extremum will not in fact be the global maximum utility. First, while this is the only point where the partial derivatives vanish, it could be a minimum or saddle point instead of a maximum. Second, the extremum may not have each $f_i \geq 0$. In either case, the true maximum utility will be found at the boundary of the allowed set of values, i.e., at least one strategy is not followed at all. In effect, such systems can be viewed as having fewer strategies.

In contrast, the equilibrium found by localized forces occurs when all strategies have the same utility so that the agents' choices remain fixed, on average. That is $\mathbf{U} = \mu \mathbf{1}$ where μ is a constant, which gives $U_{global} = \mu$. This can be combined with Eq. 10 to obtain

$$A_L \begin{pmatrix} \mathbf{f} \\ -\mu \end{pmatrix} = \begin{pmatrix} \mathbf{v} \\ 1 \end{pmatrix} \tag{15}$$

with

$$A_L = \begin{pmatrix} M & \mathbf{1} \\ \mathbf{1} & 0 \end{pmatrix} \tag{16}$$

whose solution gives the equilibrium state found by local choices, again provided $f_i \geq 0$ (otherwise, some strategies, with relatively low utilities, will not be followed in equilibrium).

4.2 The Existence of Social Dilemmas

To see that social dilemmas can be expected quite frequently, consider a typical situation in which a strategy's utility decreases with its popularity, has some fixed payoff, but does not depend on the frequency distribution among the other strategies. That is, $U_i = -f_i/a_i + v_i$ with $a_i > 0$ so that M is a diagonal matrix with $m_{ii} = -1/a_i$. The global maximum, from Eq. 13, is at $f_i/a_i = \sigma - \frac{1}{2}(\bar{v} - v_i)$, with utilities $U_i = -\sigma + \frac{1}{2}(\bar{v} + v_i)$. In these expressions we defined $\sigma = 1/\sum a_k$ and $\bar{v} = \sigma \sum a_k v_k$ as a weighted average of the components of $\mathbf{v}$. This gives

$$U_{global} = -\sigma + \bar{v} + \frac{1}{4}\sum a_i(\bar{v} - v_i)^2. \tag{17}$$

By contrast, the local equilibrium from Eq. 15 is $f_i/a_i = \sigma - \bar{v} + v_i$, with utility $U_i = -\sigma + \bar{v}$, which is also the value of U_{global} at this equilibrium. (In both cases, a negative resulting value for f_i indicates strategy i isn't used and the calculation should be repeated without it to find the correct strategy distribution).

Thus, the local equilibrium is globally optimal only if all the v_i are the same (as was the case in a limited previous study of the 3–resource case for a computational ecosystem [19].) Otherwise, the local equilibrium point will exhibit a broader strategy distribution than is optimal. Consequently, even with strategy utilities that depend only on their own frequency, the public goods social dilemma arises.

In the general case with non-diagonal matrix M, we can see that dilemmas will also often arise. In order for Eqs. 13 and 15 to give the same value for $\mathbf{f}$, we must have

$$(A_G^{-1} - A_L^{-1}) \begin{pmatrix} -\mathbf{v} \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ -\lambda + \mu \end{pmatrix} \tag{18}$$

which is a linear equation for $\mathbf{v}$. In general, most vectors $\mathbf{v}$ will not satisfy this equation, and hence the global minimum and the local equilibrium will be at different points. Thus, we can expect the results, shown explicitly above for simple diagonal matrices, to apply to a wide range of linear cost functions.

Consider, for instance, the instantiation of Braess' Paradox in a computational ecosystem, as presented in Section 3.2. There, the linear mapping that takes the frequency distribution among strategies onto utilities is given by the non-diagonal matrix

$$M = \frac{1}{2}\begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 1 & 1 & 0 \end{pmatrix}, \ \mathbf{v} = -\frac{3}{2}\begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}. \tag{19}$$

Another simple example with two strategies and a non-diagonal relation between utilities and choice frequencies is given by

$$M = -\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}, \ \mathbf{v} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}. \tag{20}$$

The average group utility attains its maximum of $-3/4$ when $f = (1/2, 1/2)$. However, the local equilibrium occurs at $f = (0, 1)$ with average utility -1. From Eq. 18 the dilemma will be avoided only in the special case where $v_2 = v_1 - 1$.

4.3 General Utility Function

For the more realistic case of nonlinear utility functions, the analysis cannot give a complete characterization of the dynamics (e.g., there may be local maxima), but can still provide some insight. The dilemma will exist (in addition to any problems due to local maxima) when the local dynamics drives the system away from the global maximum, even if it is initially quite close.

Consider the more limited problem: if the system manages to get near the global maximum, will the agent decisions tend to drive it toward the maximum or not? That is, we ask the same question as before but only about the local neighborhood of the global maximum. To study the neighborhood of the maximum, we linearize the utility function about the distribution of strategies that yields the maximum. The results for the linear case will then indicate the behavior we can expect within this neighborhood.

Suppose $\mathbf{U} = F(\mathbf{f})$ is the general relation between strategy distribution and utilities, and that the maximum of the global utility $U_{global} = \mathbf{U} \cdot \mathbf{f}$ is at

$\mathbf{f}_0$. Consider the behavior of the system near this point, i.e., for $\mathbf{f} = \mathbf{f}_0 + \mathbf{e}$ where the components of the vector $\mathbf{e}$ are small and sum to zero. This gives an expansion for the utility of various choices as

$$\mathbf{U} \approx F(\mathbf{f}_0) + F'(\mathbf{f}_0)\,\mathbf{e}. \tag{21}$$

This expression has the same form as the linear case if we take $M = F'(\mathbf{f}_0)$ and $\mathbf{v} = F(\mathbf{f}_0)$, except for a different constraint: component values sum to zero rather than one. We get an indication of whether this global maximum is also the local equilibrium by evaluating the condition for all costs to be equal: if different from the maximum, this means that agent dynamics will move away from the maximum.

How important the dilemmas will be in such cases will depend on how easily the system becomes stuck in local maxima. That is, if there are many local maxima, they may prevent the system from getting near the global maximum at all and hence the existence of a dilemma that prevents convergence to the global maximum will rarely affect the dynamics. On the other hand, systems with fewer local maxima, or that have techniques to avoid or overcome them, might be limited instead by a social dilemma. This suggests that an interesting question for future work is to determine the interplay between these effects, and whether strategies that alleviate one problem may instead increase the other.

5 Addressing Social Dilemmas

Having seen that social dilemmas can arise in a variety of situations, a natural question is how their effect can be mitigated while still retaining the autonomous local decision-making of the computational societies.

One way to circumvent public goods dilemmas, commonly used in human societies, is to introduce mechanisms that enforce contribution, such as taxation. This can be used to artificially change the relative utilities perceived by the agents when making choices. Fig. 7, for example, shows the effect of decreasing the utility of the over-used strategy (or, equivalently, increasing the resource cost) by x for the computational ecosystem described by Eq. 20.

While effective, this method requires finding the correct value for the adjustment, which may be difficult, especially in a complex, changing situation. Hence one would like the agents themselves to make the necessary adjustments. One such method is suggested by a different approach to the dilemma in human societies: create new markets that, at least partially, restrict benefits to those that contribute. This amounts to privatizing the public good

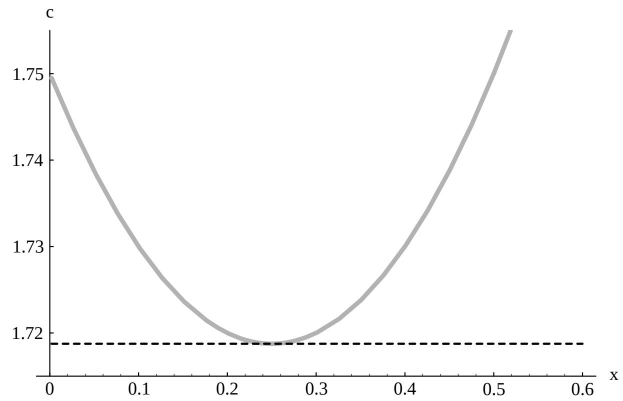

Figure 7: Removing the social dilemma from a computational ecosystem. This shows C_{global} (which equals $-U_{global}$) at the local equilibrium point selected by the agents as a function of the adjustment added to the system. The gray curve is for a cost x added to the evaluation of the under-utilized resource 2. The horizontal axis for the two curves gives the value of x. With no adjustment, the cost is 1.75. This method enables the system to achieve the minimum cost (at $x = 0.25$) where the curve meets the dashed line showing the global minimum cost.

using computational pricing and accounting mechanisms [7] within computational economies [3]. The resulting prices also provide information on competing uses for resources in a manner similar to that in market economies [22]. For example, in Fig. 6 agents could be required to pay for the use of resource 3 thus encouraging more of them to use resources 1 and 2 instead.

Finally, cooperation can be maintained if the benefits are concentrated in small groups with long-range plans [23] or if interactions are repeated [24, 25]. There are also a variety of mechanisms to evolve cooperation in such cases by selecting for cooperative behaviors whose benefits are concentrated in relatively small groups [26].

6 Discussion

In summary, additional resources can paradoxically decrease overall performance when the following basic ingredients exist: locally optimized choices, the sharing of benefits among the agents, and strategies whose utilities depend on their density distribution among the population. It is important to note that the dilemmas associated with public goods can arise even in situations where the agents share the same overall goals if their decisions are based on limited, local information. In addition to the illustrations we presented, there are a variety of other computational societies with a single overall goal that have these ingredients.

Coevolutionary systems, where the costs facing an individual depend inherently on the behavior of others, are one example. These systems arise naturally in biological and economic contexts and are increasingly appearing in the computational arena. For example, GA's can help a robot learn in specific environments through the evolution of a set of behavioral rules [27]. When a distributed GA is applied to a society of robots [28, 29], the process becomes coevolutionary. For instance, the robots might need to learn strategies for dividing tasks among themselves, so that all of the robots do not flock to the highest priority task. Another example is given by classifier systems [30] that reward agents through an analogy with a bucket brigade: each agent contributing to the final behavior receives a reward, and each of these passes some of the reward to those other agents that helped it produce that behavior. In some studies of manufacturing processes involving bucket brigades, improving the efficiency of some agents can degrade overall performance [31]. Thus we can expect the social dilemmas reported here to occur in some instances of classifier systems as well.

Another example is cooperative problem solving in which information obtained by an agent working on one part of the problem is passed to others [32, 33, 34]. For instance, the crossover operator of GA's, which takes two "parent" agents and recombines them to produce new agents, can be interpreted as an exchange of hints. Hints can be very useful in cooperative problem solving, but often require agents to "explore" possibilities that do not directly produce solutions in order to obtain useful hints. Moreover, the benefit of a new hint depends on which other hints have already been uncovered. In such cases, the benefit of an agent's choices depends on those made by others.

These problem solvers can also include learning mechanisms such as chunking [35] which automatically add extra links in a problem space [36] representing the newly learned operations. While this often improves performance [37], density-dependence can arise if the choice of which link to follow in a particular instance is made based on how useful it was in the past as measured by the number of other agents that used it. In this situation, a dilemma could arise during learning, resulting in unexpected decrease in performance.

A computational ecology example is given by recent proposals for flexible manufacturing [38]. These introduce distributed decision-making using auctions to allocate tasks to machines. In these cases agents must determine the best machine to use, given the nature of its task and the current load on the machines.

Finally, density-dependence might be intentionally introduced to improve a computational method. For instance, many search algorithms face the problem of premature convergence to a single state of low, but not optimal, cost. To avoid this, steps are taken to maintain a diverse set of agents. In GA's, this can be done by increasing the mutation rate, but this random variation also destroys useful structures. More sophisticated methods for GA's [39, 40] and other methods [41] rely on explicitly keeping the agents apart in solution space. This can be viewed as a density-dependent modification to the cost function, i.e., the desirability for a state depends on how many other agents are already "near" it. In this case, the cost function is defined by the configuration of the agents themselves, giving a coevolutionary system.

These examples show that density-dependence can arise from the nature of the problem itself or be introduced intentionally into the solution method. The resulting coevolutionary systems, in which autonomous agents attempt to coordinate their activities, can face social dilemmas in a wide range of situations. Fortunately, as in human societies, these social dilemmas can be alleviated by a number of mechanisms, such as taxation, privatization, and the establishment of norms. The need to distinguish between situations in which computational social dilemmas are harmful from those in which they might instead be beneficial suggests another direction for further research.

7 Acknowledgments

We thank Stephen Freeman for examples of solved social dilemmas in computational contexts.

References

[1] Bernardo A. Huberman and Tad Hogg. The behavior of computational ecologies. In B. A. Huberman, editor, *The Ecology of Computation*, pages 77–115. North-Holland, Amsterdam, 1988.

[2] Les Gasser and Michael N. Huhns, editors. *Distributed Artificial Intelligence*, volume 2. Morgan Kaufmann, Menlo Park, CA, 1989.

[3] Carl A. Waldspurger, Tad Hogg, Bernardo A. Huberman, Jeffery O. Kephart, and W. Scott Stornetta. Spawn: A distributed computational economy. *IEEE Trans. on Software Engineering*, 18(2):103–117, February 1992.

[4] Michael P. Wellman. A market-oriented programming environment and its application to distributed multicommodity flow problems. *J. of Artificial Intelligence Research*, 1:1–23, 1993.

[5] Tad Hogg. Dynamics of multiagent systems. A World Wide Web page with URL ftp://parcftp.xerox.com/pub/dynamics/multiagent.html, 1994.

[6] G. Hardin. The tragedy of the commons. *Science*, 162:1243, 1968.

[7] Mark S. Miller and K. Eric Drexler. Markets and computation: Agoric open systems. In B. A. Huberman, editor, *The Ecology of Computation*, pages 133–176. North-Holland, Amsterdam, 1988.

[8] D. E. Rumelhart, J. L. McClelland, and the PDP Research Group, editors. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition.* MIT Press, Cambridge, Mass, 1986.

[9] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, 1983.

[10] M. R. Garey and D. S. Johnson. *A Guide to the Theory of NP-Completeness.* W. H. Freeman, San Francisco, 1979.

[11] Natalie S. Glance and Tad Hogg. Dilemmas in computational societies. In Lesser V, editor, *Proc. of the 1st International Conference on Multiagent Systems (ICMAS95)*, pages 117–124, Menlo Park, CA, June 1995. AAAI Press.

[12] J. H. Holland. *Adaptation in Natural and Artificial Systems.* University of Michigan Press, Ann Arbor, MI, 1975.

[13] Stephanie Forrest. Genetic algorithms: Principles of natural selection applied to computation. *Science*, 261:872–878, August 13 1993.

[14] Joel E. Cohen and Paul Horowitz. Paradoxical behavior of mechanical and electrical networks. *Nature*, 352:699–701, 1991.

[15] A. D. Irvine. How Braess' paradox solves Newcomb's problem. *International Studies in the Philosophy of Science*, 7(2):141–160, 1993.

[16] John Grefenstette. A user's guide to GENESIS version 5.0. Technical report, Navy Center for Applied Research in AI, Naval Research Laboratory, 1990.

[17] Bernardo A. Huberman and Natalie S. Glance. Evolutionary games and computer simulations. *Proceedings of the National Academy of Science USA*, 90:7716–7718, August 1993.

[18] Stephanie Forrest and Melanie Mitchell. What makes a problem hard for a genetic algorithm? Some anomalous results and their explanation. *Machine Learning*, 13:285–319, 1993.

[19] J. O. Kephart, T. Hogg, and B. A. Huberman. Dynamics of computational ecosystems. *Physical Review A*, 40:404–421, 1989.

[20] Tad Hogg. Social dilemmas in computational ecosystems. In *Proc. of the 1995 Intl. Joint Conf. on Artificial Intelligence (IJCAI95)*, Menlo Park, CA, 1995. AAAI Press. to appear.

[21] Richard Dawkins. *The Selfish Gene.* Oxford Univ. Press, Oxford, 1976.

[22] Friedrich A. Hayek. Competition as a discovery procedure. In *New Studies in Philosophy, Politics, Economics and the History of Ideas*, pages 179–190. University of Chicago Press, Chicago, 1978.

[23] Natalie S. Glance and Bernardo A. Huberman. Dynamics of social dilemmas. *Scientific American*, 270(3):76–81, March 1994.

[24] R. Axelrod and W. Hamilton. The evolution of cooperation. *Science*, 211:1390–1396, March 1981.

[25] Jonathan Bendor and Dilip Mookherjee. Institutional structure and the logic of ongoing collective action. *American Political Science Review*, 81(1):129–154, March 1987.

[26] Herbert A. Simon. A mechanism for social selection and successful altruism. *Science*, 250(4988):1665–1668, December 21 1990.

[27] G. Spencer. Automatic generation of programs for crawling and walking. In Jr. K. E. Kinnear, editor, *Advances in Genetic Programming.* MIT Press, Cambridge, Mass., 1994.

[28] Eiichi Horiuchi and Kazuo Tani. Architecture and implementation issues about learning for a group of mobile robots with a distributable genetic algorithm. *Journal of Mechanical Engineering Laboratory*, 47(6):247–256, 1993.

[29] J. R. Rush, A. P. Fraser, and D. P. Barnes. Evolving cooperation in autonomous robotic systems. In *Proceedings of the IEEE International Conference on Control*, London, March 21-24 1994. Institute of Electrical Engineers.

[30] J. H. Holland. A mathematical framework for studying learning in classifier systems. *Physica*, 22D:307–317, 1987.

[31] John J. Bartholdi, III and Donald D. Eisenstein. A production line that balances itself. *Operations Research*, 1995. to appear.

[32] Scott H. Clearwater, Bernardo A. Huberman, and Tad Hogg. Cooperative solution of constraint satisfaction problems. *Science*, 254:1181–1183, November 22 1991.

[33] Tad Hogg and Colin P. Williams. Solving the really hard problems with cooperative search. In *Proc. of the 11th Natl. Conf. on Artificial Intelligence (AAAI93)*, pages 231–236, Menlo Park, CA, 1993. AAAI Press.

[34] Pedro S. de Souza and Saroush Talukdar. Asynchronous organizations for multi-algorithm problems. In *Proc. of ACM Symposium on Applied Computing (SAC93)*, Feb. 1993.

[35] J. E. Laird, P. S. Rosenbloom, and A. Newell. Chunking in soar: The anatomy of a general learning mechanism. *Machine Learning*, 1:1, 1986.

[36] A. Newell and H. Simon. *Human Problem Solving*. Prentice-Hall, Englewood Cliffs, NJ, 1972.

[37] J. Shrager, T. Hogg, and B. A. Huberman. A graph-dynamic model of the power law of practice and the problem-solving fan-effect. *Science*, 242(4877):414–416, 1988.

[38] David M. Upton. A flexible structure for computer controlled manufacturing systems. *Manufacturing Review*, 5(1):58–74, 1992.

[39] Michael L. Mauldin. Maintaining diversity in genetic search. In *Proc. of the 1984 Natl. Conf. on Artificial Intelligence*, Los Altos, CA, 1984. William Kaufmann.

[40] Robert E. Smith, Stephanie Forrest, and Alan S. Perelson. Searching for diverse, cooperative populations with genetic algorithms. *Evolutionary Computation*, 1(2):127–149, 1993.

[41] Peter Shell, Juan Antonio Hernandez Rubio, and Gonzalo Quiroga Barro. Improving search through diversity. In *Proc. of the 12th Natl. Conf. on Artificial Intelligence (AAAI94)*, pages 1323–1328, Menlo Park, CA, 1994. AAAI Press.