

Effort Profiles in Multi-Agent Resource Allocation

H. Van Dyke Parunak
ERIM

PO Box 134001
Ann Arbor, MI 48113-4001
1-734-623-2509

vparunak@erim.org

Sven Brueckner
ERIM

PO Box 134001
Ann Arbor, MI 48113-4001
1-734-623-2529

sbrueckner@erim.org

John Sauter
ERIM

PO Box 134001
Ann Arbor, MI 48113-4001
1-734-623-2513

jsauter@erim.org

Robert Savit
University of Michigan

Physics Department
Ann Arbor, MI 48109
1-734-764-3426

savit@umich.edu

ABSTRACT

Multi-agent systems are particularly appropriate for resource allocation, but configuring them for efficient operation requires understanding their dynamics. Concepts from statistical physics, such as phase transitions, can help. In decision problems such as constraint satisfaction, such transitions exhibit an easy-hard-easy effort profile, so that highly overconstrained problems are easier to solve than those near the transition. The conventional wisdom is that the profile in optimization problems such as resource allocation is monotonic, becoming more difficult as constraints increase. Contrary to this lore, we exhibit an easy-hard-easy profile in a multi-agent resource allocation problem. We compare problems that exhibit such a profile with others that do not and offer insights as to when such behavior can be expected and why it is desirable from a practical perspective.

Keywords

Complexity, Phase Transition, Phase Change, Resource Allocation, Negotiation, Effort Profile

1. INTRODUCTION

The modular, dynamic, and ill-structured nature of resource allocation problems makes them natural candidates for multi-agent systems. Typically, each agent represents a different task (consumers) or resource (supplier) in allocation negotiations. These systems must often operate in real time, so it is important to configure and operate them to avoid computational bottlenecks.

For thirty years [4], computer scientists have understood that some problems are intrinsically complex, in the sense that general solutions by deterministic algorithms require time that is at least exponential in the length of the input, and thus unattainable in reasonable time for realistic instances. These problems include many that are of great practical importance [7], including a wide variety of planning, scheduling, and resource allocation scenarios.

More recently, it has been recognized that many instances of such “NP-hard” problems are in fact tractable, and that the boundary that separates tractable from intractable instances is

mathematically similar to a phase transition in statistical physics [2, 10]. Most of the theoretical work underlying these results is based on analysis of a decision problem, the satisfiability problem (SAT), in which the objective is to determine whether or not there exists a set of assignments to variables that satisfies a Boolean expression in conjunctive normal form. The transition has been observed both in backtracking approaches [13] and in some local search scenarios [3, 21], though the underlying mechanisms that drive the transition are different in the two cases.

We are interested in how phase changes of this sort manifest themselves in multi-agent resource allocation systems. These systems are optimization problems rather than decision problems. That is, we are not interested in deciding whether or not a perfect solution exists, but rather in finding assignments to problem variables that maximize the utility of our (usually imperfect) solution.

That optimization problems can have phase transitions is not novel. In fact, physical systems that inspire the phase transition metaphor are naturally viewed as solving an optimization, not a decision, problem (e.g., minimizing a Hamiltonian). The conventional wisdom [8, 23] is that these two classes of problems differ qualitatively in the behavior of the computational effort needed to solve a problem as the degree of constraint increases. In decision problems the effort is greatest at the transition, and lower for both over- and under-constrained problems (“easy-hard-easy,” or EHE). However (it is asserted), effort increases monotonically with constraint for optimization (“easy-hard,” or EH). In fact, the pattern depends not only on the nature of the problem being solved, but also on the solution method. We exhibit an optimization problem (specifically, resource allocation) that shows both patterns (and others), depending on the solution method. A phase *transition* may be characteristic of a *problem*, independent of solution method, but effort *profile* is tied to the *process*.

Section 2 reviews previous research on phase transitions in both decision and optimization problems. Section 3 reports an easy-hard-easy profile in a simple model of resource allocation. Section 4 reports the behavior of solution effort in some other resource allocation systems that we have examined. Section 5 discusses our findings, and Section 6 concludes.

2. REVIEW OF PHASE TRANSITIONS IN COMPUTATION

For the past decade, computer scientists have been applying the concept of a phase transition from statistical mechanics to computational processes. [8, 9] offer convenient reviews. Three results set the background for our investigations.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

AAMAS '02, July 15-19, 2002, Bologna, Italy.

Copyright 2002 ACM 1-58113-000-0/00/0000...\$5.00.

2.1 Backtracking Search in 3SAT

[13] observed an important feature of the distribution of 3SAT problems as the ratio of clauses to variables increases. When the ratio is less than a critical value $R_c \approx 4.27$, virtually all randomly generated 3SAT problems are satisfiable, while for higher ratios virtually none are. The percentage is 50% at R_c . (Thus the transition itself is a function only of problem structure, not of process.) The effort required to solve an instance (i.e., declare it either satisfiable or unsatisfiable) is measured by the number of calls to the Davis-Putnam resolution procedure, and is low for ratios far from R_c , but peaks at R_c , thus EHE.

A simple intuition explains this behavior. In under-constrained problems, many different assignments to the variables can satisfy an expression, and the search procedure will typically stumble on one fairly quickly. If the problem is over-constrained, there will be many inconsistencies among its variables, and the again the search procedure will discover one quickly. Near R_c , many partial assignments can be extended almost completely before determining whether or not they will satisfy the formula, requiring extensive backtracking and raising the computational cost. This intuition relies on the complete nature of backtracking search. In every case, the problem is proven to be either satisfiable or unsatisfiable, and the difficulty of solution depends on how easy the proof is to achieve.

[13] speculate, “We suspect that our results on hard and easy areas generalize to *all* SAT procedures, but this remains to be seen” (italics theirs). Our results support the “but.”

2.2 Local Search in 3SAT

The intuition explaining the EHE profile in backtracking search relies on proving a formula either satisfiable or unsatisfiable, so it does not apply to heuristics. However, the same profile has been detected in heuristic approaches to 3SAT. Since heuristics cannot prove a problem unsolvable, the experiments are performed on a population of problems known to be solvable. It is remarkable that 1) computational effort peaks, and 2) it does so at the same R_c where Davis-Putnam peaks on a population not preselected for solvability. The hill-climbing heuristic used is:

1. Begin with a random assignment of variables.
2. Change one variable assignment at a time to hill-climb toward a better assignment.
3. Rerandomize if it falls into a local minimum.

Since all problems explored are solvable, the algorithm runs until it finds the solution, and the number of restarts (including the initial start) functions as the measure of computational effort

It is reasonable that this algorithm should succeed sooner with underconstrained problems than with those that are somewhat more constrained, but not immediately apparent why over-constraint should also yield lower computational costs. [21] has recently developed an alternative intuition that explains this behavior. The nemesis of any hill-climbing algorithm is the “local minimum” (step 3), which traps the search and can be escaped only by a nonlocal and usually random move to some other part of the search space. Such a minimum, together with the adjacent states that lead to it, forms a basin of attraction, and a local search mechanism that stumbles into such a basin will be drawn to the local minimum. Yokoo explores a large number of solvable 3SAT problems of varying degrees of constraint with respect to the number and size of these basin. He shows that the average width of basins decreases monotonically as constrainedness increases,

and that the number of basins increases up to the point of the phase transition, then levels off. The result is that beyond the transition point, a solvable problem offers fewer deceptions to the problem solver, and computational effort decreases.

It is important to notice that the distribution of problems that Yokoo addresses is different from that under backtracking search. Yokoo selects problems from the universe of satisfiable formulae, while backtracking search selects them from the universe of all formulae. There are $a = {}_{2k}C_n$ possible clauses of n literals each over a universe of k variables, so the total number of possible problems with c clauses is $a C_c$. This value, increases nearly exponentially in c for fixed n and k , but the fraction of satisfiable formulae decreases monotonically with c .

2.3 Optimizing Search

Decision problems are binary: either the problem is solved completely, or it is declared unsolvable. The world is often less hospitable, leading to optimization problems. In this model, every result is a solution, but each solution is associated with a scalar quantity that varies monotonically with how desirable the result is. The quantity may be framed as a cost to be minimized, or a utility to be maximized. Resource allocation problems are often better viewed as optimization problems than decision problems. In practice, resources are almost always limited, and one seeks the best possible allocation of limited resources across tasks, not a binary decision that gives up because a perfect solution is unavailable.

One model for such a search is a randomly generated tree of fixed depth, with costs associated with each edge. The net cost of each node (and in particular, the leaf nodes) is the sum of the edge costs between the root and that node. The task is to find a leaf node with lowest cost. Studies of this model (e.g., [11, 22]) show that the complexity of the problem is represented by bp , where b is the mean branching factor of the tree and p is the probability that an edge has cost 0. Thus bp is the expected number of children having the same cost as their parent. The number of nodes of such a tree that must be generated to find a minimal leaf shows a phase transition at $bp = 1$. For $bp < 1$, the complexity is exponential in the depth of the tree, while for $bp \geq 1$, it is polynomial in tree depth. Thus the computational effort increases monotonically with problem complexity. The traveling salesman problem [24] and an optimizing version of 3SAT [23] show the same pattern. Thus it has been suggested [8] that phase transitions in optimization problems are characterized by an EH profile rather than the EHE profile of decision problems. But these claims are cautious. “In short, the phase transitions of some NP-complete decision problems have easy-hard-easy patterns and the phase transitions of some NP-hard optimization problems follow easy-hard patterns. These phase transition results exhibit a discrepancy between the phase transitions of decision and optimization problems” [23]. The second sentence urges the reader to accept a general principle, but the repeated “some” in the first sentence recognizes the possibility of such exceptions as we exhibit.

3. CONSTRAINT-DRIVEN PHASE TRANSITIONS IN A RESOURCE ALLOCATION GAME (RAG)

The AORIST project [16] is exploring a simple model of multi-agent resource allocation, the “resource allocation game.” Full

details of this Resource Allocation Game (RAG) are available elsewhere [19]. This game has several salient features common to many multi-agent resource allocation problems.

Consumers and suppliers.—At each turn of the game, N consumers select among G suppliers. In the results reported here, $G = 2$, although similar results obtain for higher G . Each supplier has the same capacity, and the sum of all supplier capacities is C . The relative values of N and C drive the dynamics described below. If $N \ll C$, we say that resources are abundant. If $N \gg C$, they are scarce, and for $N \approx C$, they are limited.

Consumer utility.—Consumers care whether their needs are met. Each consumer receives one point if the total number of consumers n choosing its supplier is less than or equal to C/G , in which case the supplier is said to be underloaded. If the supplier is overloaded, we explore two different ways of rewarding the consumers, each of which is appropriate in different application domains. In the *binary satisfaction* approach, consumers on an overloaded supplier receive no points. For example, if a power generation circuit is overloaded, a circuit breaker may open, and none of the loads will be able to operate. In the *partial satisfaction* approach, each consumer on an overloaded supplier receives $C/(Gn) < 1$ point, so that the supplier’s capacity is distributed evenly across its consumers. In an alternative power scenario, each load may receive less than its desired amount of current. The detailed results in this paper obtain for binary satisfaction, but the main features are robust and occur for partial satisfaction as well. From a system perspective, one wishes to maximize the total reward to all consumers (or equivalently, the average reward across consumers).

Experience.—Many resource problems are solved repeatedly. Vehicles seeking spare parts, project managers seeking staff and funds, and telecommunications packets seeking channels repeat these activities over and over. In our game, each consumer learns the state of each supplier after each turn. It chooses its next supplier based on the supplier state vectors in the previous m turns.

Choice function.—Each consumer makes its choice deterministically. We model this decision process with a look-up table, or “strategy,” mapping from vectors of m system states to supplier choice. Denote an underloaded supplier by + and an overloaded one by -. For $G = 2$, possible states at each turn are (+,+), (+,-), (-,+), and (-,-). (+,+) is only accessible if $N \leq C$, and (-,-) is only accessible if $N > C+1$. Each consumer knows the state of all suppliers, but other aspects of this game are invariant using more local information [5, 15], and we expect the results reported here to generalize similarly.

Learning.—Consumers learn from their experience. Each consumer has two randomly generated strategies. At each turn of the system, a consumer rewards each of its strategies with the award that the consumer would have received on that turn if it had used that strategy. In selecting its next move, the consumer chooses that strategy with the currently higher aggregate score.

As the consumers play successive turns, they adjust the ranking of their strategies. Under certain conditions, they learn to prefer strategies that are maximally distinct from those of the other consumers, thus distributing themselves over the suppliers in a way that maximizes system performance. Several metrics are useful in studying this behavior. We typically run a given configuration for 10,000 turns, and compute metrics on the last 1000 turns, thus focusing on the results of the learning process rather than

transient dynamics during learning. (Some metrics, not discussed here, are computed over the entire run.) Useful metrics include:

The variance of the load on one supplier, normalized by N : σ^2/N . When this parameter is high and $N < C$, average agent wealth tends to be low, since wide swings in supplier load open up the possibility for capacity on one supplier to be wasted while the other is overloaded. The quantity σ^2 is related to the Hamiltonian of the system [12].

The average wealth across the agents, reflecting the overall performance of the system. While not as foundational as σ^2/N , this metric is heuristically useful in assessing system performance.

The number of rows in their strategy tables that consumers actually access. This measure reflects the computational load on the agents. Our strategies are a surrogate for more complex decision processes. Whatever the process, we expect that it will require more reasoning as the consumer encounters more states of the m -turn history. In practice, we generate the rows of each strategy table as they are needed, so our measure of computational load is simply the number of rows in the strategy table that an agent has generated by the end of the game.

Previous research has revealed the existence of a phase transition in σ^2/N as a function of m in the configuration when $N = C+1$, known as the “minority game” [19]. (Actually, as we show in a forthcoming paper, a more general independent variable is the entropy of the history of system states). In this paper, we explore the behavior of our metrics when we vary N and C relative to one another. This performance landscape corresponds more closely to the actual conditions under which resource allocation takes place.

Figure 1 shows the behavior of the average agent wealth as a function of N and C . The logistic shape of the curve is characteristic of phase transitions, and in this case the transition lies close to the line $N = C$. This result is hardly surprising. When resources are abundant, even random choices would distribute the consumers so that they rarely overload a supplier. When resources are scarce, consumers are hardly ever satisfied. Near $N = C$, the system must transition between these extremes.

Figure 2 shows σ^2/N over the same space. This surface is much more complex, showing seven distinct regions. At the extremities ($N \gg C$ or $C \gg N$) are two regions in which σ^2/N is fairly smooth as a function of N and C . As we move toward the diagonal, we pass into areas in which the dependence of σ^2/N on C and N is rougher. Moving further toward the diagonal from either direction, σ^2/N decreases and has a smoother dependence

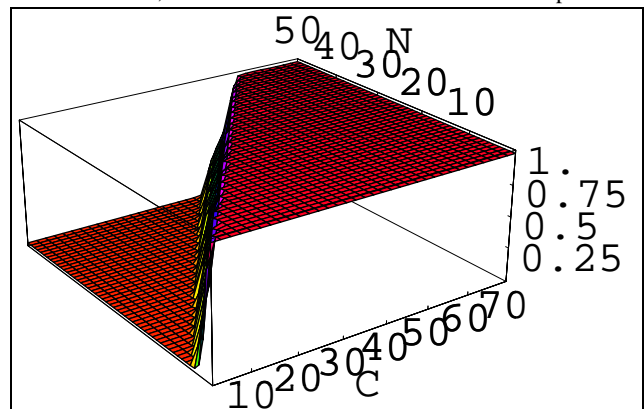


Figure 1: Average agent wealth.—Overall system performance drops dramatically around $N = C$. Each point is the average of 13 runs of 10k turns; $m = 6$.

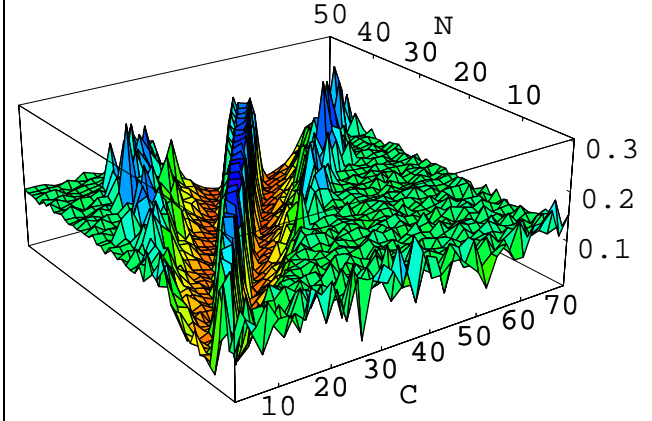


Figure 2: Mean variance in supplier load.—This parameter shows a much more complex landscape, and peaks around the phase transition.

on C and N . Finally, very close to the diagonal σ^2/N increases, reaching its maximum value near $C=N$.

These seven regions correspond to qualitatively different behaviors on the part of the consumers [18]. In this discussion, we focus on the region near $N = C$, where σ^2/N peaks. Figure 3 shows a slice at constant $C = 200$ through this peak. (We show only the flank on the side where $N > C$; a theorem proven in [17] shows that for fixed N , given an initial set of strategies to N agents, the group choices made by those agents are identical in the two games played with those agents and with $C=N-1-d$ and $C=N-1+d$ for any integer d , and qualitatively the same symmetry obtains when we hold C constant and vary N on either side.) This figure plots the result of each run of the system separately, in contrast with Figure 2, which plots the average at each value of N and C . In the region corresponding to the flanks of the central peak in Figure 2, the values of σ^2/N for different runs separate into two quite distinct groups. For a given C and a range of N , there is a well-defined coexistence region between two very distinct phases in these resource allocation games, comparable to the coexistence of different phases in a physical system. The values of σ^2/N in Figure 2 at the top of the central peak and at the bottom of the neighboring valleys accurately reflect typical behavior of individual runs. But the average values along the flanks are atypical; any given individual run belongs either to the high or the low phase. This system thus exhibits not one but two phase changes, one on each flank of the system.

What is the distribution of computational effort as one moves through this phase change? Figure 4 plots the average number of rows of the strategy table generated for each value of N along the same $C = 200$ line used in Figure 3. This value peaks just over the coexistence region identified in Figure 3, and drops away elsewhere. The inset (for a system with $C=60$) shows that it is as low for $N \gg C$ as it is for $N \ll C$. In other words, this game is an optimization problem, but exhibits a gross EHE profile (with fine structure we do not discuss here) around $N = C+1$.

We can develop an intuition supporting this pattern of computational load. The peak near the minority game, and the valleys on either side, represent qualitatively different dynamics in the game, resulting from differing numbers of states accessible to the system. At $N = C+1$, by construction, only $(+,-)$ and $(-,+)$ are accessible, and near $N \approx C+1$,

they are the most common system states. There are enough consumers that it is unlikely all will be satisfied, and few enough that it is unlikely that both suppliers will be overloaded. As one moves in to the valleys, the probability increases that the system will enter states $(-,-)$ (when $N > C$), or $(+,+)$ (when $N < C$). The ability of a given population of agents to distinguish these states depends on the distribution of strategies (generated in our case randomly and fixed for the duration of the game). Agents can effectively distribute themselves over resources only when their strategies differ on the state histories of length m that they experience. In the valley, the balanced states (say, $(+,+)$) are common enough that they will appear in several distinct state histories, and the strategies held by the agents in almost any population differ on some of these histories, so the agents can learn to accommodate them as well as the unbalanced states. On the peak, the balanced states are so rare that no agents need to recognize them. But on the flanks, the balanced states are rare, and only a few distinct histories include them. Some populations have strategies that largely overlap on these few histories, thus cannot discriminate the new balanced states, and so continue to behave as though they were at the top of the peak, forming the upper phase seen in Figure 3. Others are sensitive to the new states, and allocate themselves in a way that is qualitatively similar to games played in the valleys, forming the lower phase. Furthermore, a given population must work harder as it moves into the valley to reach an appropriate distribution of its members across resources, as reflected in the increased level of computational effort.

In other words, the critical feature driving the phase transition is the diversity among the agents and their strategies. Some of this diversity is frozen into the agents through the randomly generated strategies, and some of it emerges as the agents dynamically learn to prefer one strategy to another and so distribute themselves over the problem space. The emergent dynamics behave differently on the peak and in the valley, and in between they take longer to converge.

4. BEHAVIOR OF OTHER RESOURCE ALLOCATION SYSTEMS

While the resource allocation game captures many features of resource allocation, we do not claim that it subsumes every possible resource allocation scheme. The question naturally arises

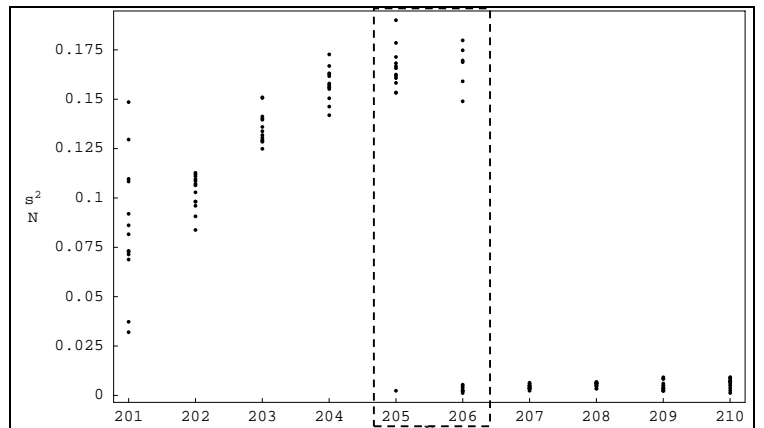


Figure 3: Mean load variance detail.—This plot shows each of the 13 runs at each $N \in \{201, 210\}$ for $C = 200$, $m = 6$. At $N = 205$ and $N = 206$, instances of the system are distributed across two distinct states, comparable to the coexistence of two physical phases (e.g., ice and water).

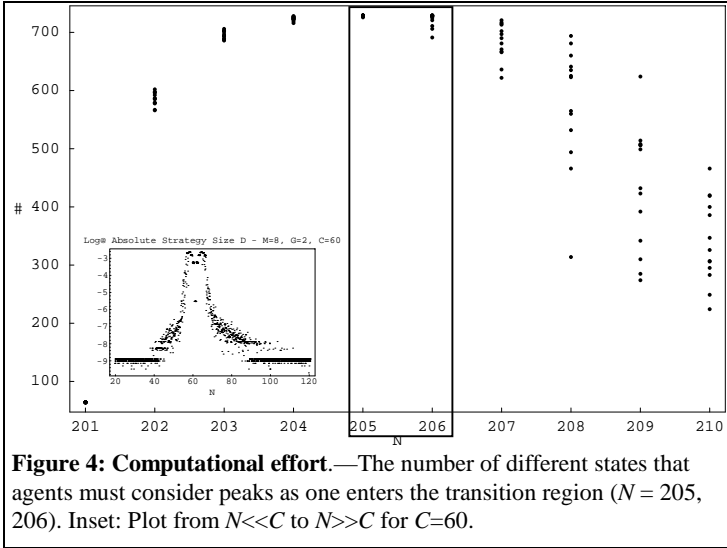


Figure 4: Computational effort.—The number of different states that agents must consider peaks as one enters the transition region ($N = 205, 206$). Inset: Plot from $N \ll C$ to $N \gg C$ for $C=60$.

whether other resource allocation systems exhibit the same pattern that we observe as load varies with respect to capacity. The short answer is that some do, and some do not.

The experiments reported in this section are preliminary. We are grateful to the creators of these systems for giving us access to them while they are still under development. Due to the complexities of experimenting with other peoples' code, we do not claim to have explored the behavior of these systems completely, and in some cases more refined experimental configurations might yield more complex effort profiles. These limitations in fact emphasize one of our main points: computational effort in optimization problems is a function not only of the abstract structure of the problem, but also of the process used to solve it, and adjusting the process may be expected to change the profiles.

4.1 UKansas

Agents in the Case-Based Reflective Negotiation Model (CBRNM) at the University of Kansas [20] base negotiations on their past experience, as stored in a case base. In the application we studied, two agents negotiate over access to CPU time. The initiating agent keeps sending facts (evidence) to convince the responding agent that it should give up some CPU usage. The requested amount is defined by the parameter $CPUShortage$. The responding agent weights each fact and adds the weighting to its internal evidence level. This level represents the amount of CPU the responding agent is willing to give up. If this amount becomes larger than the requested amount, the responding agent agrees to give up the requested resources, up to its $maxCPUGiveUp$ threshold. If the initiating agent requests more than $maxCPUGiveUp$, the responder makes a counter offer. If the negotiation runs over the allotted time, it is aborted and no resource is exchanged.

In this series of experiments, all negotiations run to completion (no timeout). The $maxCPUGiveUp$ threshold of the receiving agent is set to 10%, so all negotiations for less than this threshold reach a 100% success rate. But the effort (number of messages exchanged) increases as the amount of CPU requested moves closer to the threshold, since it takes more evidence to convince the responding agent. Above the threshold, the number of messages required to convince the responding agent is constant, because the sequence of facts does not vary in this experiment. We define the success rate as the ratio of the granted resource and the

requested resource. Above $maxCPUGiveUp$, the expected success rate is $maxCPUGiveUp/cpuShortage$. For all settings of the $cpuShortage$ parameter, there is enough time and enough facts available to convince the responding agent to give up the requested resource. We never reach a timeout failure and we never see a counter offer caused by insufficient evidence.

Figure 5 shows performance and effort in this simple scenario. From the structure of the problem, the distinguished point where one might expect phase-transition-like behavior is when $cpuShortage = maxCPUGiveUp$ (0.1). Though there is no sharp transition, as load increases with respect to capacity, the quality of solutions decreases, and the amount of computation needed (measured by messages exchanged) increases.

4.2 CAMERA Initial System

The CAMERA system at ISI [14] allocates resources to training missions for Marine aviators. Each mission has a number of requirements, each of which can be satisfied by a number of resources. The resources are bombing ranges, pilots, instructors, and aircraft. The most constrained resource, empirically, is the set of ranges.

The CAMERA team has developed a number of resource allocation algorithms. The one currently being transferred to the Marines is a greedy system, which works as follows.

1. Each mission bids first for range time, which is the most constrained resource. It queries the ranges for their available time slots. It receives a list of all the time slots that the range director has made available, including those that have already been committed to other missions. It then marches through all available time slots in 30-minute start intervals looking for a time slot of sufficient length that is not already committed.
2. After securing a range time slot it identifies the pilots requiring that training, qualified instructors, and aircraft qualified for the mission type. It then initiates three parallel requests for commitment (RFC's):
 - a. The first pilot in the ordered list of pilots
 - b. All qualified instructors
 - c. All qualified aircraft
3. From the instructors and aircraft it accepts the first to respond and decommits to all the rest that commit. For the pi-

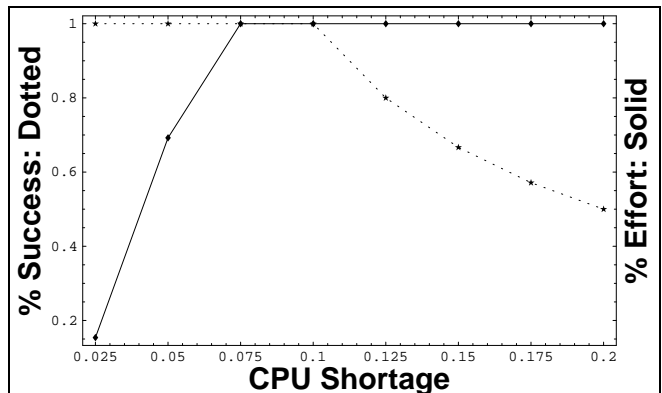
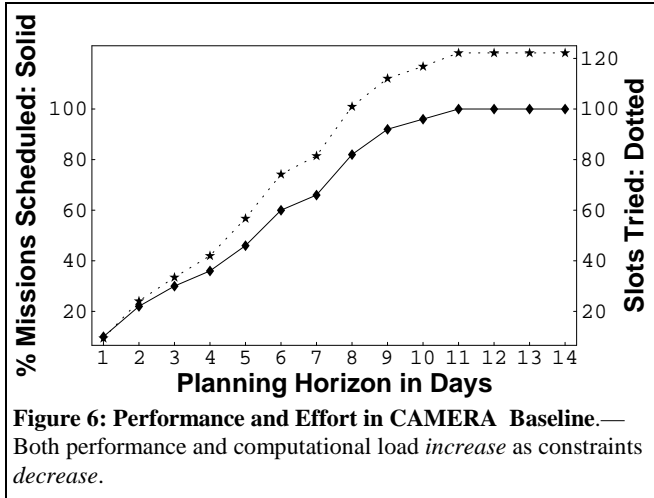


Figure 5: Performance and Effort in Case-Based Negotiation.—Number of messages increases, and success decreases, as $MaxCPUGiveUp$ parameter increases.



lot, it will send an RFC to each pilot in turn until it finds the first pilot who can commit.

4. If it cannot find any pilot or aircraft, or instructor to commit to that time slot, it decommits all commitments and samples the next 30-minute start time on the range.

In this system, each task requires multiple types of resources (pilots, instructors, aircraft, and ranges), and resources take the form of time slots. As a convenient way to adjust the relative load and capacity of the system, we hold the number of missions constant at 50, and vary the time horizon over which the missions could be scheduled. In practice the planning horizon is constrained by when the missions need to fly, but for our experimental purposes, we ignore these constraints and focus simply on the percentage of the 50 missions that can successfully be scheduled.

Figure 6 shows the dependency of system performance (% of missions scheduled) and computational effort (the number of slots that have to be tried) on the planning horizon. The performance metric resembles that in the UKansas system: the less constrained the system (the longer the horizon), the higher the performance, with no sharp transition. However, unlike either UKansas or the RAG, the effort also increases with the planning horizon. In terms of movement from high to low performance, the profile is neither EH (as in the UKansas system) nor EHE (as in decision problems and our system), but HE (“hard-easy”)! The least work is needed for the shortest planning horizon, when the system is most tightly constrained. The system’s greedy algorithm assembles sets of resources for each mission sequentially, and does not explore alternate configurations if the initial configuration fails. Thus the effort expended, like the system performance, is monotonic in system capacity. We speculate that if the missions selected range slots randomly rather than using the greedy approach, there would be fewer collisions among missions vying for the same time slots and the effort would decrease as the planning horizon increased, resulting in an EHE profile.

4.3 CAMERA Marbles Schemes

The CAMERA team is investigating a family of more sophisticated bidding mechanisms known as Marbles [6], applied to the same air

mission scheduling domain. These schemes are a subset of single-resource auctions, with two interesting characteristics. First, they schemes resemble Edgeworth barter rather than Walrasian auctions [1], in that resources move among consumers as the protocol executes, rather than waiting for all bidders to submit bids and then clearing once for all. This ability of resources to move from one task to another in the course of a negotiation enables the system to search the space of possible allocations more thoroughly than does the original scheme described in Section 4.1 above, and bears some similarity to the iterated assignments in AORIST. Second, when an individual task senses that it will be unable to get the resources it needs, it stops trying, thereby dynamically reducing the constraints on the system.

At least one member of the Marbles set of algorithms, called Marblesize, does exhibit an EHE profile, as measured by the number of messages that must be exchanged (Figure 7). As in the Kansas and early CAMERA systems, the performance curve (the upper curve in both figures) does not show the discontinuity typically associated with a phase transition, but there is a drop-off in computational cost beyond the peak, resulting from the surrender of tasks that are not succeeding. By removing themselves, these tasks reduce the overall demand on the system, and the remaining tasks have an easier time meeting their requirements.

5. DISCUSSION

The class of resource allocation schemes that we study in this paper is significantly more complex and less schematic than the abstract decision and optimization problems on which most work on computational complexity has been done to date. For this reason, comparison of the two sets of results requires considerable care. Some useful conclusions can be drawn.

5.1 Profiles without Transitions

Most studies of effort profiles focus in the vicinity of a “phase transition.” To the physicist, this expression has a very specific meaning: a point of non-analyticity across which there is a qualitative change (e.g., from zero to non-zero) in some system parameter (the “order parameter”). Such formal transitions have been identified in a variety of decision problems (e.g., 3SAT) and optimization problems (e.g., random tree search), as well as in the RAG as a function of m in the minority configuration ($N = C + 1$). One signature of a phase transition is a sudden shift in an order parameter in the general shape of a logistic curve, as in **Figure 1**.

While the RAG does show such a discontinuous shift in performance, the systems in Section 4 (at least as configured in our experiments) do not. Yet they do have different load profiles, and

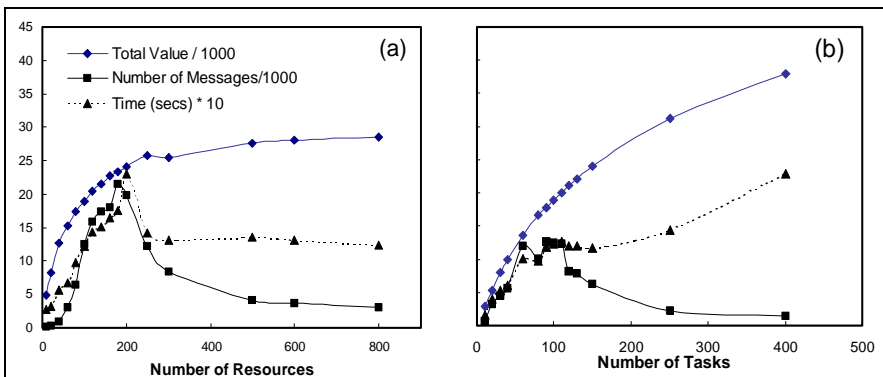


Figure 7: EHE profiles for Marblesize. (a) 100 tasks, (b) 100 resources

these differences can be of great practical importance in fielding an operational system. These results suggest the importance of examining the load profile of a resource allocation system whether or not it manifests a true phase transition.

More generally, our results emphasize the importance of two distinctions: problem from process, and phase transition from profile. Some problems (such as 3SAT) have phase transitions that depend solely on the structure of the problem, independent of the solution process, but effort profiles can vary depending on the solution process.

5.2 Computational Load in Optimization

Clearly, some optimization problems, including some resource allocation algorithms (the Kansas system) do show EH profiles. Just as clearly, the resource allocation game, early CAMERA, and Marbles do not. One can draw two morals.

1. Conclusions from abstract models may not always hold for realistic systems.
2. More importantly, a single *problem* (e.g., that handled by the early CAMERA system and Marbles) may show very different *profiles* (HE or EHE), depending on the *algorithm* used. Computational load in optimization depends not only on the problem, but also on the process used to solve it.

5.3 New Intuitions

The experimental evidence we have examined suggests two intuitions concerning computational load in resource allocation. The first of these applies to the structure of the problem, while the second calls attention to the nature of the process.

First, the state space accessible to a population of suppliers can differ drastically between a region where supply is approximately the same as demand and regions where supply and demand are unbalanced. As a system shifts into and out of the balanced region, one may expect phase changes that result in a peaking of computational load. This intuition is similar to that which explains the EHE profile in local search in 3SAT. In both cases, dramatic changes in computational load are correlated with changes in the state space.

Second, contrary to earlier speculation [13], the process does matter. We speculate that algorithms that attempt to distribute demand across resources through learning and other forms of diversification in the population of consumers are more likely to exhibit EHE profiles than those that do not. Simpler algorithms show EH or HE behavior. Adaptability is seen both in our RAG system and in Marbles. In Marbles, the system adapts to problem state as individual tasks decide to surrender, thus changing the load and the system's dynamics. The RAG models agent adaptation in two ways. First, agents learn which of their strategies to prefer, and this learning proceeds differently depending on the state of the system. Second, the portion of the strategy that an agent uses varies depending on where in the state space the system is operating. For example, in the highly overloaded region, the system is almost always in state $(-, -)$. On average half of the agents will have strategies that are tied for strings of state $(-, -)$ of any length m . As a result, their strategies will be tied, and the agents will randomly choose between them at each step. However, in regions where more states are available, this symmetry between strategies will be broken, and the agents as a group will exhibit deterministic behavior. Thus the agents exhibit load-dependent behavior. (We emphasize that our strategy mechanism is not in-

tended to represent a realistic decision process, but rather serves as a probabilistic model for more complex decision behavior.)

This latter observation is particularly appropriate to a multi-agent system. A centralized architecture encourages the use of the same process at each point in the problem, even if the process as a whole is sensitive to the state of the problem. In a MAS, each agent can adapt differently, enabling the system to respond not only to the overall state of the problem but also to local variations. The resulting variability in the process is much greater than in a centralized architecture, and (we posit) is more likely to exhibit EHE profiles in problems whose state spaces permit them.

5.4 Practical Importance

Discussions of computational complexity often rest on an underlying assumption that phenomena such as phase transitions are a necessary evil, an undesirable disturbance that a responsible system designer should seek to avoid. When the transition shows an EHE profile, this assumption should be qualified, because the peak of the profile can give valuable information and even guide process design.

Effective configuration and management of a system that performs resource allocation (say, a maintenance facility or an inventory management system) require assessing the overall capacity of the system. In simple resource allocation problems such as our RAG, it is trivial to define the capacity of the system: if each consumer needs one unit of resource and there are C units available, the capacity of the system is just C . Real systems have "effective capacities" that may not be so easy to compute. For example, the capacity of a resource may depend on its state of maintenance and thus its history; resources may be shared among multiple systems for which an integrated representation does not exist; or the load on a resource may vary nonlinearly and even nonmonotonically with the number of consumers. In such cases, the capacity of the system cannot be determined analytically, but must be estimated by observation. Such estimation can be attempted from a plot of the system's performance as a function of demand, analogous to **Figure 1**, and when the transition is sharp as in this figure, this information is adequate. However, the performance transition at capacity is often not this sharp, as a result of factors such as finite system size, noise, or even the lack of a formal phase transition, as in the systems discussed in Section 4. It would be difficult to define the capacity of these systems based on their performance curves. When a system exhibits an EHE profile, the point of nonmonotonicity provides an estimate of the system's effective capacity that can be used to tune and adjust it.

In fact, it may be worthwhile using existence of an EHE profile as a design guide for optimization processes. Our results suggest that the presence of such a profile reflects the adaptivity of a process, so that EHE processes may be able to reach performance levels not accessible to EH process.

6. CONCLUSION

The trade-off between system performance and computational cost is one of the major engineering concerns in information technology, and is particularly critical with highly nonlinear systems (such as MAS's) that can exhibit phase transitions. The load profiles of such systems as they are more tightly constrained depend both on the underlying problem and on the process used to solve it. Optimization systems can exhibit not only EH profiles, but also the EHE profiles more commonly associated with decision systems. Even absent a formal phase transition, analyzing these pro-

files is an important in understanding a system's performance. In particular, EHE profiles reflect adaptability in the behavior of the problem-solving agents, and are useful in establishing the effective capacity of a complex resource allocation system.

7. ACKNOWLEDGMENTS

This work is supported in part by the DARPA ANTS program, contract F30602-99-C-0202 to ERIM, under DARPA PM Janos Sztipanovits and Vijay Raghavan. The views and conclusions in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the US Government. We gratefully acknowledge the cooperation of Bob Neches, Alejandro Bugacov, and colleagues at ISI in experiments on the CAMERA system, and of Costas Tsatsoulis and his colleagues at the University of Kansas for providing an instrumented version of their system.

8. REFERENCES

- [1] R. Axtell and J. Epstein. Distributed Computation of Economic Equilibria via Bilateral Exchange. Brookings Institution, Washington, DC, 1997.
- [2] P. Cheeseman, B. Kanefsky, and W. M. Taylor. Where the really hard problems are. In *Proceedings of IJCAI-91*, pages 331-337, Morgan Kaufmann, 1991.
- [3] D. Clark, J. Frank, I. Gent, E. MacIntyre, N. Tomov, and T. Walsh. Local Search and the Number of Solutions. In *Proceedings of Second International Conference on Principles and Practices of Constraint Programming (CP-96)*, pages 119-133, 1996.
- [4] S. A. Cook. The complexity of theorem-proving procedures. In *Proceedings of Third Annual ACM Symposium on Theory of Computing*, pages 151-158, 1971.
- [5] M. A. R. de Cara, O. Pla, and F. Guinea. Learning, competition, and cooperation in simple games. *The European Physical Journal*, B 13(R419), 2000.
- [6] M. Frank, A. Bugacov, J. Chen, G. Dakin, P. Szekely, and B. Neches. The Marbles Manifesto: A Definition and Comparison of Cooperative Negotiation Schemes for Distributed Resource Allocation. In *Proceedings of AAI Symposium on Negotiation Methods for Autonomous Cooperative Systems*, AAI, 2001.
- [7] M. R. Garey and D. S. Johnson. *Computers and Intractability*. San Francisco, CA, W.H. Freeman, 1979.
- [8] C. P. Gomes, T. Hogg, T. Walsh, and W. Zhang. IJCAI-2001 Tutorial: Phase Transitions and Structure in Combinatorial Problems. 2001. HTML and PDF, <http://www.cs.wustl.edu/~zhang/links/ijcai-phase-transitions.html>.
- [9] T. Hogg, B. Huberman, and C. Williams, Editors. *Frontiers in Problem Solving: Phase Transitions and Complexity. Special Issue of Artificial Intelligence*, vol. 81, issue 1-2 (March). Netherlands, Elsevier, 1996.
- [10] T. Hogg, B. A. Huberman, and C. Williams. Phase Transitions and the Search Problem. *Artificial Intelligence*, 81:1-15, 1996.
- [11] R. M. Karp and J. Pearl. Searching for an optimal path in a tree with random costs. *Artificial Intelligence*, 21:99-117, 1983.
- [12] M. Marsili, D. Challet, and R. Zecchina. Exact solution of a modied El Farol's bar problem: Efficiency and the role of market impact. 1999. PDF File, http://ttt.lanl.gov/PS_cache/cond-mat/pdf/9908/9908480.pdf.
- [13] D. Mitchell, B. Selman, and H. Levesque. Hard and Easy Distribution of SAT Problems. In *Proceedings of AAAI-92*, 1992.
- [14] R. Neches and P. Szekely. CAMERA Project: Coordination and Management Environments for Responsive Agents. 1999. Web Page, <http://www.isi.edu/camera/>.
- [15] M. Paczuski, K. E. Bassler, and Á. Corral. Self-organized Networks of Competing Boolean Agents. *Physics Review Letters*, (forthcoming), 2000.
- [16] H. V. D. Parunak. Agents Overcoming Resource-Independent Scaling Threats (AORIST Project). 2000. Web Page, www.erim.org/cec/projects/aorist/.
- [17] R. Savit, S. A. Bruckner, H. V. D. Parunak, and J. Sauter. General Structure of Resource Allocation Games. ERIM, Ann Arbor, MI, 2001.
- [18] R. Savit, S. A. Bruckner, H. V. D. Parunak, and J. Sauter. Phase Structure of Resource Allocation Games. *Phys. Rev. Lett.*, (forthcoming), 2001.
- [19] R. Savit, R. Manuca, and R. Riolo. Adaptive Competition, Market Efficiency, and Phase Transitions. *Physical Review Letters*, 82(10):2203-2206, 1999.
- [20] C. Tsatsoulis. Case-Based Reflective Negotiation Model. 2001. Web page, <http://www.itc.ku.edu/ANTS/>.
- [21] M. Yokoo. *Distributed Constraint Satisfaction: Foundations of Cooperation in Multi-Agent Systems*. Berlin, Springer, 2001.
- [22] W. Zhang. *State Space Search: Algorithms, Complexity, Extensions, and Applications*. New York, Springer, 1999.
- [23] W. Zhang. Phase Transitions and Backbones of 3-SAT and Maximum 3-SAT. In *Proceedings of 7th International Conference on Principles and Practice of Constraint Programming (CP2001)*, 2001.
- [24] W. Zhang and R. E. Korf. A study of complexity transitions on the asymmetric traveling salesman problem. *Artificial Intelligence*, 81:223-239, 1996.