A Game-Theoretic Formulation of Multi-Agent Resource Allocation

Jonathan Bredin[†], Rajiv T. Maheswaran[‡], Çagri Imer[‡], Tamer Başar[‡], David Kotz[†], and Daniela Rus[†]

ABSTRACT

This paper considers resource allocation in a network with mobile agents competing for computational priority. We formulate this problem as a multi-agent game with the players being agents purchasing service from a common server. We show that there exists a computable Nash equilibrium when agents have perfect information into the future. From our game, we build a market-based CPU allocation policy and a strategy with which an agent may plan its expenditures for a multi-hop itinerary. We simulate a network of hosts and agents using our strategy to show that our resource-allocation mechanism effectively prioritizes agents according to their endowments and that our planning algorithm handles network delay gracefully.

1. INTRODUCTION

Mobile-agent systems allow programs to autonomously relocate from one host to another. An agent may jump to one site to filter a database, jump to another site to access a camera, and to a third to process the results of the previous two hops before returning the results to the user. For each of the three hops, there may be alternative sites to access substitutable resources and this choice of execution location subjects hosts to greater congestion volatility.

Code mobility is a software architectural feature that has many benefits [13]. Many user programs can reduce the effect of network latency by relocating their execution states to be closer to their data. Mobility is a flexible abstraction that can speed software development and deployment. Additionally, mobility provides an extra layer of fault tolerance.

To regulate mobile agents and provide resource owners with greater incentive to host agents, we present a market-based system in which agents bid for computational priority from hosts. We propose that agents use a system that creates incentives for hosts to participate, provides agents with the costs of their actions, and allows agents of heterogeneous priorities to operate simultaneously.

We construct a resource-allocation policy where hosts take bids from agents for prioritized access to computational resources (CPU time). The rate at which an agent computes is proportional to its bid relative to the sum of all current bids at the host. Hosts collect revenues from each agent at a rate equal to the agent's bid.

We apply this policy to a mobile-agent system with several different types of resources distributed throughout a network. Each agent has a sequence of resources to consume (an itinerary) and an endowment of electronic currency to be used to purchase resource access to complete its itinerary.

We formulate the hosts' resource-allocation problem as a game with the players being agents competing for a resource from a common server. We show how to compute the unique positive Nash equilibrium explicitly under perfect information when there are two or more players to minimize an agent's serial execution time. Starting from this simple mechanism and an assumption of perfect knowledge, we develop an optimal agent bidding strategy that plans an agent's expenditure of multi-task itineraries. Our bidding strategy minimizes execution time while preserving a prespecified budget constraint. We complete our work by presenting a simulation of mobile agents competing for computational access in a network of heterogeneous hosts.

The paper is organized as follows. In Section 2, we describe the system model. In Section 3, we derive the optimal bid for a single agent's first job given load statistics. In Section 4, we show the existence of a Nash equilibrium that can be computed by the server, when all agents submit bid functions of a given form. In Section 5, we show that the Nash equilibrium obtained in the previous section is unique. In Section 6, we simulate a network where agents submit the optimal policies to servers that allocate resources according to the resulting Nash equilibrium. The results are further discussed in Section 7, and some related work is described in Section 8. The paper ends with some concluding remarks included in Section 9 and an Appendix.

2. SYSTEM MODEL

The system model follows that presented in [6]. We consider a network graph where agents are generated at some subset of nodes. Each agent is given a task of completing a set of jobs of different types in a given sequence by purchasing resources from service providers located throughout

[†]Department of Computer Science, Dartmouth College, 6211 Sudikoff Lab, Hanover, NH 03755
[‡]Coordinated Science Laboratory, University of Illinois, 1308 W. Main Street, Urbana, IL 61801

the network. The i-th agent begins with an endowment of I_i dollars to spend to complete its task and wishes to minimize the total time taken to complete a sequence of jobs given its budget constraint. There is no additional utility for an agent to have a positive endowment after completing all of its jobs.

We assume that there are K types of service and that each agent only needs to complete a job of a particular type at most once. The agent's task can be represented as the sequence $\{q_k^i\}_{k=1}^K$, where q_k^i is the size of the k-th type of job for the i-th agent, and $q_k^i = 0$ implies that the agent's task does not include completing a job of type k. We assume that there are several service providers for each type of service, and the capacity of the provider chosen by the i-th agent to complete its job of type k is c_k^i. We make many of the assumptions for the sake of notational simplicity, and the following analysis can easily be extended to the more general case when agents must complete a job type multiple times.

A host provides its entire capacity at no cost if only one agent is requesting service. If more than one agent is currently requesting service, the capacity is partitioned as follows. The i-th agent receives service at a rate proportional to its bid relative to the sum of all bids,

$$v_k^i = c_k^i \left(\frac{u_k^i}{u_k^i + \theta_k^{-i}} \right), \tag{1}$$

where u_k^i is the amount (in dollars per second) that the i-th agent bids for service, the provider receives bids totaling θ_k from the set of agents, $\mathcal{J}_k$, and $\theta_k^{-i} = \sum_{j \in \mathcal{J}_k, j \neq i} u_k^j$. Thus, if the service rate is constant, the time taken by the i-th agent to complete its job of type k is q_k^i / v_k^i, i.e.,

$$t_k^i = \frac{q_k^i (u_k^i + \theta_k^{-i})}{c_k^i u_k^i}, \tag{2}$$

and the expenses are $t_k^i u_k^i$, i.e.,

$$e_k^i = \frac{q_k^i (u_k^i + \theta_k^{-i})}{c_k^i}. \tag{3}$$

3. SINGLE AGENT OPTIMIZATION

The problem facing the i-th agent is how to choose its bids, $\{u_k^i\}_{k=1}^K$. Computationally, this can be formulated as an optimization problem to minimize the total time of completing its task, $T_i = \sum_{k=1}^K t_k^i$, such that the budget constraint is preserved, $E_i = \sum_{k=1}^K t_k^i u_k^i \leq I_i$. At this point, we assume that the agent spends at a constant rate u_k^i at the provider for the k-th type of job, and also that θ_k^{-i} is known, independent from u_k^i and remains constant throughout the time that server is being utilized. In this section, since we are dealing with the i-th agent only, to simplify our derivation of the bidding procedure, we drop the i superscripts and subscripts in all our variables except θ_k^{-i}. We retain the notation of θ_k^{-i} to differentiate the sum of bids at the host submitted by competing agents from θ_k, the sum of all bids including the i-th agent's.

We have the following optimization problem:

$$\min_{u_k} \sum_{k=1}^K t_k \qquad \text{s.t.} \qquad \sum_{k=1}^K e_k \leq I. \tag{4}$$

We solve this problem using Lagrangian methods. We first introduce the Lagrangian:

$$\mathcal{L} = \sum_{k=1}^K t_k + \lambda \left(\sum_{k=1}^K e_k - I \right). \tag{5}$$

Substituting for $t_k = t_k^i$ and $e_k = e_k^i$ into equation (5) and taking partial derivatives with respect to u_k, we obtain

$$\frac{\partial \mathcal{L}}{\partial u_k} = \frac{-q_k \theta_k^{-i}}{c_k u_k^2} + \lambda \frac{q_k}{c_k} = 0 \qquad \Rightarrow \qquad \lambda = \frac{\theta_k^{-i}}{u_k^2}. \tag{6}$$

Note that $\theta_k^{-i} > 0$ implies $\lambda > 0$ for all but the trivial case when only one agent bids. Thus we have the following relationship between any two bids, j and k:

$$u_k = u_j \sqrt{\frac{\theta_k^{-i}}{\theta_j^{-i}}}. \tag{7}$$

Incorporating the inequality constraint, we get

$$\lambda \frac{\partial \mathcal{L}}{\partial \lambda} = \lambda \left(\sum_{k=1}^K \frac{q_k (u_k + \theta_k^{-i})}{c_k} - I \right) = 0. \tag{8}$$

Since $\lambda > 0$, it follows that the inequality constraint must be satisfied with equality. Substituting for $\{u_k\}_{k=2}^K$ in terms of u_1 using the relationship in equation (7), we have

$$\frac{q_1}{c_1}(u_1 + \theta_1^{-i}) + \sum_{k \neq 1} \frac{q_k}{c_k} \sqrt{\frac{\theta_k^{-i}}{\theta_1^{-i}}} u_1 + \sum_{k \neq 1} \frac{q_k}{c_k} \theta_k^{-i} - I = 0 \tag{9}$$

Solving the previous equation for u_1, we finally get

$$u_1 = \frac{I - \sum_{k \neq 1} \frac{q_k}{c_k} \theta_k^{-i} - \frac{q_1}{c_1} \theta_1^{-i}}{\frac{q_1}{c_1} + \sum_{k \neq 1} \frac{q_k}{c_k} \sqrt{\frac{\theta_k^{-i}}{\theta_1^{-i}}}}, \tag{10}$$

which yields the bid for the first job for the i-th agent given the loads of the servers for the jobs in its sequence.

4. MULTI-AGENT NASH EQUILIBRIUM

Consider now, without loss of generality, a provider with capacity of resource type 1, and N agents desiring service. We assume that the agents have some estimate of future loads, θ_k^{-i}, so that they are able to estimate how much of their money to spend for this current job. Such an estimate can be obtained by augmenting the system with "advertising" agents that periodically update and make available the values of total bids at various servers throughout the network. A simple estimate of θ_k^{-i} would be the means of the loads of all the servers of type k jobs. Given a load estimate for future jobs, the optimal bid from equation (10) is:

$$u_1^i = f_i(\theta_1^{-i}) := \frac{\alpha^i - \beta^i \theta_1^{-i}}{\beta^i + \frac{\gamma^i}{\sqrt{\theta_1^{-i}}}}, \tag{11}$$

where

$$\alpha^i := I - \sum_{k \neq 1} \frac{q_k^i}{c_k^i} \theta_k^{-i}, \tag{12}$$

$$\beta^i := \frac{q_1^i}{c_1^i}, \tag{13}$$

$$\gamma^i := \sum_{k \neq 1} \frac{q_k^i}{c_k^i} \sqrt{\theta_k^{-i}}. \tag{14}$$

Intuitively, α^i represents the estimate of the money available for the current job. If that amount is less than zero, the agent cannot afford to purchase service under the current state of the network. We require that the bids be nonnegative. We have $\beta^i > 0$, $\theta_k^{-i} > 0$, and $\gamma^i \geq 0$, with equality only if the agent has one job. If $\alpha^i \leq 0$, f_i will return a negative value. Thus the i-th agent will only submit a bid if $\alpha^i > 0$.

At the server, we would like to generate a set of bids that forms a Nash equilibrium [3] with respect to the policies of the N agents:

$$\{u_1^i = \max\{0, f_i(\theta_1^{-i})\}\}_{i=1}^N \quad . \tag{15}$$

A Nash equilibrium solution is a set of bids where no agent can gain an advantage by unilaterally changing its bid. One possibility of reaching the Nash equilibrium is to use a decentralized algorithm where each agent makes an initial bid and then updates its bid at preset time intervals, δ, using the iteration

$$u_1^i(t+\delta) = f_i(\theta_1^{-i}(t)), \tag{16}$$

where $u_i^i(t)$ and $\theta_1^{-i}(t)$ denote the i-th agent's bid and sum of competing bids, respectively, at time t, all for resource type 1. Unfortunately, this algorithm rarely converges to a Nash equilibrium and is suboptimal due to the inconsistency of the initial guesses and subsequent iterations.

Instead, we focus on a centralized method to obtain optimal bids. The agents submit bid *functions* in the form of equation (11) and the server produces the optimal bids for each agent. To formulate this new method, we translate each agent's bid function domain from θ_1^{-i} to a single common domain, θ_1. Once we change the domain, we show that the space over θ_1 is continuous under most conditions and that we can approximate the remaining instances with our existing framework.

Using u-space has a deficiency in that its dimension increases with the number of agents. To reduce our search space, we iterate over a common domain for all agents, θ_1-space, where $\theta_1 := \theta_1^{-i} + u_1^i$. Modifying the policies in equation (15), we get the following implicit relations between u_1^i and θ_1:

$$\{u_1^i = \max\{0, f_i(\theta_1 - u_1^i)\}\}_{i=1}^N \quad . \tag{17}$$

From this, we can obtain an explicit function $g_i(\theta_1) : \theta_1 \to u_1^i$. Figure 1 illustrates how $f_i(\theta_1 - u_1^i)$ shifts as θ_1 varies. Figure 2 demonstrates the shape of $g_i(\theta_1)$. Outside the range $\theta_1 \in (0, \alpha^i/\beta^i)$, $g_i(\theta_1)$ takes the value of 0. We now derive $g(\theta_1)$.

Substituting $\theta_1 - u_1^i$ for θ_1^{-i} in (11), in the range $\theta_1 \in (0, \alpha^i/\beta^i)$, we have the following:

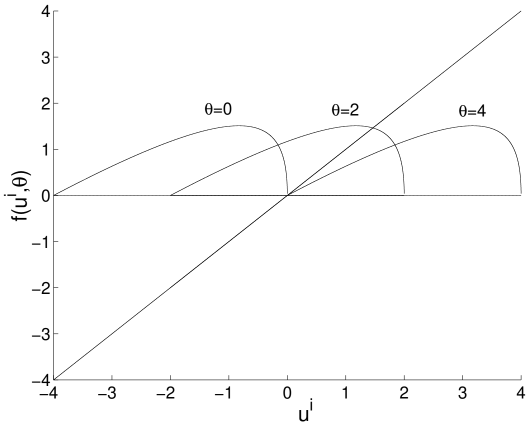

Figure 1: Behavior of $\max(0, f_i(\theta - u_1^i))$ for $\alpha^i/\beta^i = 4$

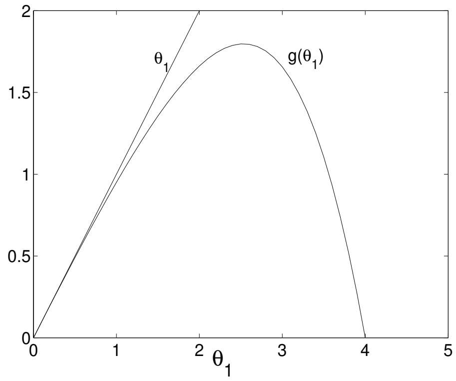

Figure 2: Form of $g_i(\theta_1)$

$$u_1^i = \frac{\alpha^i - \beta^i(\theta_1 - u_1^i)}{\beta^i + \frac{\gamma^i}{\sqrt{\theta_1 - u_1^i}}}, \tag{18}$$

which leads to a quadratic equation in u_1^i. Dropping the i superscript, we have:

$$\gamma^2 u_1^2 + (\alpha - \beta\theta_1)^2 u_1 - (\alpha - \beta\theta_1)^2 \theta_1 = 0 \tag{19}$$

Taking the positive root of the equation with respect to u_1, we have $u_1 = g(\theta_1)$, where

$$g(\theta_1) = \frac{(\alpha - \beta\theta_1)^2}{2\gamma^2} \left(-1 + \sqrt{1 + \frac{4\gamma^2\theta_1}{(\alpha - \beta\theta_1)^2}} \right) \tag{20}$$

when $\theta_1 \in (0, \alpha/\beta)$ and $u_1 = 0$ otherwise.

We see that g is continuous over the range $\theta_1 = 0$ to $\theta_1 = \alpha/\beta$. We also note that for $\theta_1 \in (0, \alpha/\beta)$:

$$\frac{\partial g}{\partial \theta_1} = \frac{2\beta(\alpha - \beta\theta_1)}{2\gamma^2} + \frac{-2\beta(\alpha - \beta\theta_1)^2 + 2\gamma^2(\alpha - \beta\theta_1) - 4\beta\gamma^2\theta_1}{2\gamma^2\sqrt{(\alpha - \beta\theta_1)^2 + 4\gamma^2\theta_1}}, \tag{21}$$

and when $\theta_1 = 0^+$,

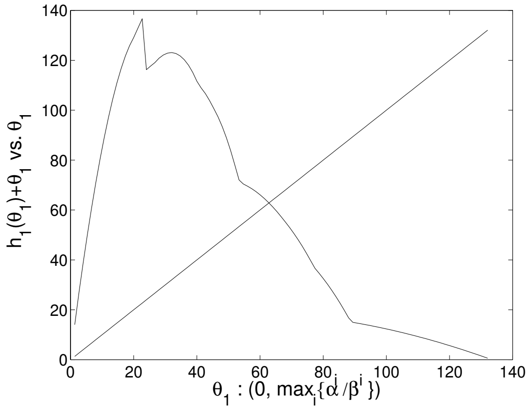

Figure 3: Sample plot of $\sum_{i=1}^{N} g_i(\theta_1)$ versus θ_1 for 16 agents

$$\left.\frac{\partial g}{\partial \theta_1}\right|_{\theta_1=0^+} = \frac{1}{2\gamma^2}\left[2\beta\alpha + \frac{-2\beta\alpha^2 + 2\gamma^2\alpha}{\sqrt{\alpha^2}}\right] = 1. \tag{22}$$

Returning to the question of the optimal bids for N agents, we seek θ_1 and $\{u_1^i\}_{i=1}^N$ that satisfy the definition $\theta_1 = \sum_{i=1}^N u_1^i$ and equation (17) for all agents. An equivalent problem is to find a value of θ_1 such that $\sum_{i=1}^N u_1^i - \theta_1 = \sum_{i=1}^N g_i(\theta_1) - \theta_1 =: h_1(\theta_1) = 0$. Because $\partial h_1/\partial\theta_1|_{\theta_1=0^+} = -1 + \sum_{i=1}^N 1 > 0$ if $N \geq 2$, we know h_1 is increasing to the right of zero and $h_1(0^+) > 0$. We also know that $h_1(\max_i\{\alpha^i/\beta^i\}) = -\max_i\{\alpha^i/\beta^i\} < 0$ for the non-trivial case where at least two agents have $\alpha^i > 0$. Because h_1 is the sum of continuous functions, h_1 is continuous as well and must be zero for some value of $\theta_1 \in (0, \max_i\{\alpha^i/\beta^i\})$. We solve for this value by using a bisection search of h_1 in the given range. A sample of $\sum_{i=1}^N g_i(\theta_1)$ versus θ_1 is depicted in Figure 3.

If an agent has only one job left to complete, it can be shown that $u_1 = \theta_1$ for $\theta \in (0, Ic/q)$ is its optimal policy, which is equivalent to having $\gamma = 0$, which violates one of the assumptions made earlier. However, by using L'Hôpital's rule, we see that

$$\begin{aligned}
\lim_{\gamma\to 0^+} g(\theta_1) &= \lim_{\gamma\to 0^+} \frac{\frac{1}{2}\frac{8\theta_1\gamma}{(\alpha-\beta\theta_1)^2}(\alpha-\beta\theta_1)^2}{4\gamma\sqrt{1+\frac{4\gamma^2\theta_1}{(\alpha-\beta\theta_1)^2}}} \\
&= \lim_{\gamma\to 0^+} \frac{\theta_1}{\sqrt{1+\frac{4\gamma^2\theta_1}{(\alpha-\beta\theta_1)^2}}} \\
&= \theta_1.
\end{aligned}$$

Thus, if we require agents with only one job to submit bid functions with $\gamma > 0$, we allow the agents to approximate their optimal solutions to arbitrary precision and still preserve the assumed structure, which yields an equilibrium solution.

5. UNIQUENESS OF EQUILIBRIUM

In this section, we prove that the Nash equilibrium whose existence was verified in the previous section is in fact unique.

Let $O_i = (0, \alpha^i/\beta^i)$ be indexed such that $O_1 \supset O_2 \supset \cdots \supset O_N$ (i.e, $\alpha^1/\beta^1 > a_1^2/\beta^2 > \cdots > \alpha^N/\beta^N$) where N is the number of agents at a server. Let us define

$$h_1^n(\theta_1) = \sum_{i=1}^{n} g_i(\theta_1) - \theta_1. \tag{23}$$

We have already shown that $h_1(\theta_1) = h_1^N(\theta_1) = 0$ has at least one solution on $O = \cup_{i=1}^N O_i = (0, \max_i\{\alpha^i/\beta^i\}) = O_1$.

Theorem 1. *$h_1^N(\theta_1) = 0$ has only one solution on O.*

To prove Theorem 1, we must first establish the concavity of $g(\theta_1)$ and prove an initial lemma. In the Appendix, we show that $(\partial^2 g_i/\partial\theta_1^2) < 0$ on O_i. From the definition of h_1^n in equation (23) and the definition of the indices, it can be seen that

$$\frac{\partial^2 g_i}{\partial\theta_1^2} < 0 \text{ on } O_i\ \forall i \quad\Rightarrow\quad \frac{\partial^2 h_1^n}{\partial\theta_1^2} < 0 \text{ on } O_n, \tag{24}$$

$$\left.\frac{\partial g_i}{\partial\theta_1}\right|_{\theta_1=0^+} = 1\ \forall i \quad\Rightarrow\quad \left.\frac{\partial h_1^n}{\partial\theta_1}\right|_{\theta_1=0^+} = n-1, \tag{25}$$

$$g_i(0) = 0\ \forall i \quad\Rightarrow\quad h_1^n(0) = 0. \tag{26}$$

Also, h_1^n is a continuous function of θ_1.

Lemma 1. *If $h(x)$ is a twice continuously differentiable function on $[r,s]$, $(\partial^2 h/\partial x^2) < 0$ on (r,s), $h(r) > 0$, and $h(s) < 0$, then there exists a unique point $x_0 \in (r,s)$ s.t. $h(x_0) = 0$.*

Proof. We prove this lemma by contradiction. The Intermediate Value Theorem states that there is at least one value $x_0 \in (r,s)$ s.t. $h(x_0) = 0$. Because $(\partial^2 h/\partial x^2) < 0$ on (r,s), we know that h is strictly concave on (r,s), i.e.,

$$ah(x) + (1-a)h(y) < h(ax + (1-a)y) \tag{27}$$

for $a \in (0,1)$ and $x, y \in (r,s)$. Suppose there are two points that satisfy $h(x) = 0$, say $x_1, x_2 \in (r,s)$, where $x_1 < x_2$. Again, using the Intermediate Value Theorem, we can show that $\exists r_0 \in (r, x_1) \subset (r,s)$ s.t. $h(r_0) > 0$. Then, we have $ah(r_0) + (1-a)h(x_2) < h(ar_0 + (1-a)x_2)$ which implies $ah(r_0) < h(ar_0 + (1-a)x_2)$. If $a = (x_2 - x_1)/(x_2 - r_0) \in (0,1)$, then we have $ah(r_0) < h(x_1) = 0$, which is a contradiction since $a > 0$. Thus, there can be at most one point where $h(x) = 0$. ∎

Proof of Theorem 1. When $n = 1$, $(\partial h_1^1/\partial\theta_1)|_{\theta_1=0^+} = 0$, and $(\partial^2 h_1^1/\partial\theta_1^2) < 0$ on O_1, which implies that $h_1^1(\theta_1) < 0$ on O_1. When $n = 2$, $(\partial h_1^2/\partial\theta_1)|_{\theta_1=0^+} = 1$ and $h_1^2(0) = 0$, thus $h_1^2(0^+) > 0$. Also, $h_1^2(\alpha^2/\beta^2) = h_1^1(\alpha^2/\beta^2) < 0$ and $(\partial^2 h_1^2/\partial\theta_1^2) < 0$ on O_2. Applying Lemma 1, we get that there is a unique point θ_0 s.t. $h_1^2(\theta_0) = 0$ on O_2. But, $h_1^2(\theta_1) = h_1^1(\theta_1) < 0$ on $O_1 \cap O_2^c$; thus θ_0 is the unique point where $h_1^2(\theta_0) = 0$ on O_1.

Uniqueness of θ_0 can be shown using an inductive argument. Assume that there is a unique point $\theta_0 < \alpha^i/\beta^i$ on O_1 where $h_1^i(\theta_0) = 0$. Also assume $(\partial^2 h_1^i/\partial\theta_1^2) < 0$ on O_i and $h_1^i(\theta_1) < 0$ on $O_1 \cap O_i^c$. Along with the continuity of h_1^i, the previous result implies the following:

$$h_1^i(\theta_1)\begin{cases} > 0 & \theta_1 < \theta_0 \\ = 0 & \theta_1 = \theta_0 \\ < 0 & \theta_1 > \theta_0 \end{cases} \tag{28}$$

Algorithm 1 `Allocate Resources for Host` k

while true do
  $t :=$ time since last arrival/departure
  for all agents i **do**
    deduct $tg_i(\theta)$ from agent i's endowment
  end for
  add new agent or remove departing agent
  for all agents i **do**
    query agent i for α, β, and γ
    use (equations (12-14)) to build $g_i(\theta)$
  end for
  search for $\theta = \sum_{i=1}^{N} g_i(\theta)$ in $(0, \max_i(\alpha_i/\beta_i))$
  for all agents i **do**
    $v_k^i := c_k g_i(\theta)/\theta$
  end for
end while

Rewriting equation (23), we have $h_1^{i+1} = h_1^i + g_{i+1}$. There are two cases to consider:

Case 1. If $(\alpha^{i+1}/\beta^{i+1}) \leq \theta_0$, then equation (28) is satisfied for h_1^{i+1} because $g_{i+1}(\theta_1) = 0$ for $\theta_1 \geq \theta_0 \geq (\alpha^{i+1}/\beta^{i+1})$ and $g_{i+1}(\theta_1) \geq 0$ for $\theta_1 \leq (\alpha^{i+1}/\beta^{i+1})$. Thus, there is a unique point θ_0 where $h_1^{i+1}(\theta_0) = 0$ on O_1.

Case 2. If $\theta_0 < (\alpha^{i+1}/\beta^{i+1}) < (\alpha^i/\beta^i)$, then we have $h_1^{i+1}(\alpha^{i+1}/\beta^{i+1}) = h_1^i(\alpha^{i+1}/\beta^{i+1}) < 0$ by equation (28). We also know $h_1^{i+1}(0^+) > 0$, because $h_1^{i+1}(0) = 0$ and $(\partial h_1^{i+1}/\partial\theta_1)|_{\theta_1=0^+} = i > 0$. Since $(\partial^2 h_1^{i+1}/\partial\theta_1^2) < 0$ on O_{i+1}, we can apply Lemma 1 and arrive at the result that there is a unique point $\hat{\theta_0}$ on O_{i+1} where $h_1^{i+1}(\hat{\theta_0}) = 0$. But since $g_{i+1}(\theta_1) = 0$ for $\theta_1 > (\alpha^{i+1}/\beta^{i+1})$, we have that on $O_1 \cap O_{i+1}$, $h_1^{i+1}(\theta_1) = h_1^i(\theta_1) < 0$. Thus, we have a unique point $\hat{\theta_0}$ where $h_1^{i+1}(\hat{\theta_0}) = 0$ on O_1. ■

6. SIMULATION AND RESULTS

In this section, we introduce an algorithm that implements the resource-allocation policy from Section 4 and we describe a simulation of the policy.

A host accepts bid functions from all agents present any time an agent arrives to or departs from the site. Agents express bids through three coefficients defined in equations (12)-(14). The host takes all bids to form the bid-response function, $g(\theta)$, and uses a bisection search to find the bidding level $\theta = g(\theta)$. Algorithm 1 sketches this operation.

We base our simulation on the Swarm simulation system [14]. In the simulation, there are 100 hosts, each providing one of eight computational services, and the capacity is determined from a truncated Gaussian random variable. The hosts comprise a network stochastically generated from the GT-ITM package [7]. In GT-ITM, a network is built from a hierarchical system of transit domains connecting stub domains. The user specifies the number and average size of domains in nodes (hosts) as well as the probability that nodes are connected within the domain.

Agents are generated at a Poisson arrival rate with uniformly determined starting location and exponentially distributed number of jobs in their itineraries. Each job size is chosen from an exponential distribution. The final parameter describing an agent is its endowment size relative to the sum of its job sizes. We generate this parameter from a truncated Gaussian random variable with positive mean. This parameter reflects the owning user's preference that the task completes quickly.

Algorithm 2 `Choose Next Site for Agent` i

$t_{min} := \infty$; $nextHost := \emptyset$
for all hosts k offering service next in itinerary **do**
  $t_k :=$ [`transferLatency:` k `from:` $currentHost$] $+ c_k g(\theta_k^{-i})/(\theta_k^{-i} + g(\theta_k^{-i}))$
  if $t_{min} > t_k$ **then**
    $t_{min} := t_k$; $nextHost := k$
  end if
end for
return $nextHost$

Once an agent is injected into the network, it must formulate a route incrementally by choosing a host for the next task after completing the previous task. For the purpose of expenditure planning, for all but the next immediate host choice, agents plan to visit hosts of average capacity c_k, and average congestion θ_k, for hosts offering the corresponding service. Agents then choose the next site to be the one that minimizes the sum of execution times, assuming no change in bidding level θ_k^{-i}, and network transfer times for the next hop. Thus, our routing algorithm is greedy and naive. We sketch its operation in Algorithm 2.

Once an agent jumps to a site, it commits to finishing the current task at that site. To complete the current task, the agent submits a bidding function conforming to the restrictions in Section 5. Whenever an agent arrives or leaves a host, the host searches for an equilibrium congestion, θ.

To test the effectiveness of our resource-allocation policy, we compare the endowment of agents with their performance. Once the network has reached a steady state, we designate seven percent of the agents injected into the system as test agents. Our test agents have identical itineraries and starting hosts, but they have differing endowment sizes, spanning two standard deviations, σ, around the mean endowment, μ. We then measure the performance of agents compared to what they would have achieved in a network with zero congestion. We compute this idealized measurement by computing a shortest path through the network where the distance between any two hosts is the sum of the network transfer times and the time an agent would take to complete a job at the host without any competition, q_k/c_k.

Figure 4 shows a plot of agent endowment versus performance relative to the ideal. The figure plots the mean performance of many agents at various endowment levels, as well as the standard deviations. There is a very strong relationship between spending and the speed at which an agent completes its itinerary. We show a χ^2 linear fit of the average performance of agents with the same endowment to demonstrate the correlation of endowment and performance. Figure 5 shows a plot of a similar scenario, but this time with agent requests overloading the network's capacity. Our resource-allocation policy ignores poor agents in the interest of having some agents complete their tasks quickly. We see that agents with less than average endowments are not serviced and that more well endowed agents are prioritized as in the previous experiment.

We would also like to ascertain the effect of agents using dated information to plan their itineraries. We measure the effect by increasing the amount of time required for an agent to jump from one host to another. This increase does not explicitly age agents' information, but it does delay agents' actions from the point in time of their reasoning. Figure 6

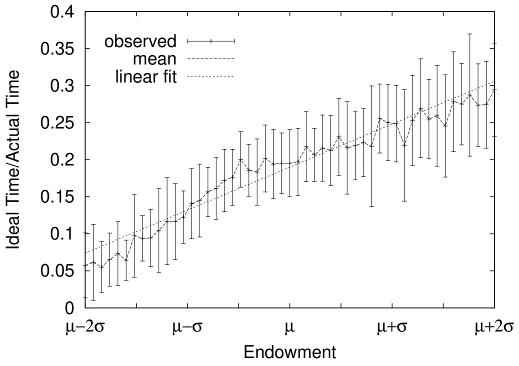

Figure 4: Endowment versus ideal time relative to actual time. We plot the observed mean and standard deviations for each endowment level.

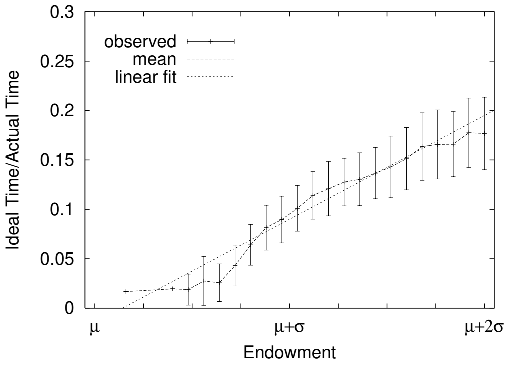

Figure 5: Endowment versus ideal time relative to actual time when agent requests exceed system capacity. Agents with endowments less than the mean are not serviced, while the rest of the population is normally prioritized by endowment.

shows how network delay affects agents' performance compared with their performance in a benchmark scenario. We see that as network delay increases, agents' performance degrades slowly.

7. DISCUSSION AND FUTURE WORK

We have presented a simple resource-allocation policy and an efficient means for agents to plan their expenditure under perfect information. Agents reformulate their expenditure plans after completing every job in their itinerary and the bidding strategies preserve agents' budget constraints. Thus, agents will always complete their itineraries. In the implementation of our simulator, however, we use discrete time units and round up when assessing agents' costs to the host, so occasionally agents cannot complete their itineraries. This occurrence is less frequent with finer time-unit granularity. Further agent failures occur when agents with small endowments cannot complete their itineraries because higher priority congestion never subsides.

Our planning algorithms rely on exactly knowing agents' job sizes. Although this assumption is reasonable for certain application domains, we would like to relax this requirement in our future work by exploring other strategies where agents have probability distributions for their job sizes instead of

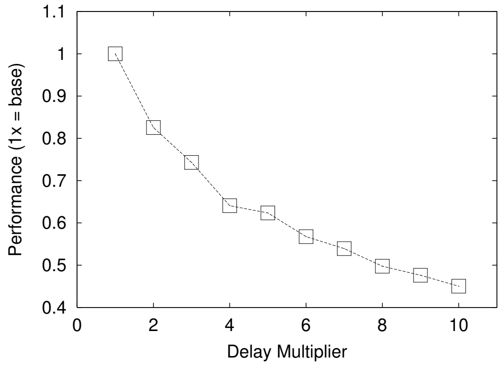

Figure 6: Agents' mean performance versus network delays relative to the mean performance in a network with a delay multiplier of one.

fixed scalar values.

In constructing our expenditure planning algorithm, we assumed that agents have already chosen a route to complete their itineraries. In the implementation, however, we can only estimate the future capacity of hosts based upon aggregate host statistics and we choose agents' routes in a greedy fashion. This technique appears to work well for a simple network topology but would fall short in a more complex network, such as a wireless network where hosts move about their environments. The problem of routing an agent through a network of hosts to minimize itinerary completion time is NP complete. To route the agent, one must choose the shortest path through a network while preserving the budget constraint and this has been shown to be NP complete through a reduction from the knapsack problem [1]. We are currently investigating using approximation algorithms to route an agent's path.

Finally, market systems require that agents have information from which to make decisions. Our future research will attempt to evaluate both the cost of distributing information, such as site congestion, as well as the cost of agents using inaccurate or dated information.

8. RELATED WORK

Our earlier work in this area adopted a currency model of resource allocation, in which agents use electronic cash to purchase needed resources and participate in an electronic market with a banking system [4]. There are several methods to determine the price of a certain resource in a given market structure. We investigated the derivation of the equilibrium pricing scheme in a single seller market with a number of buyers, having a Cobb-Douglas type utility function [5]. In more recent work, we determine prices in a market structure in which the servers provide a price curve for the resources that they sell, and we obtain the equilibrium price as the result of a utility maximization problem [16].

We are not the first to research the possibility of using market-based control for mobile-agent systems. Telescript supported a system where agents had permits whose strength diminished over time and as they moved around the network [19]. The weakening of permits approximates currency exchange. The Geneva Messengers mobile-agent system includes support for agents to buy CPU priority, memory, and network access [18]. Neither system, however, explores how agents should plan in market-based environments.

POPCORN is a market-based system where users submit Java programs to a centralized server that contracts the programs' executions out to servers [17]. User programs in POPCORN, however, are not autonomously mobile and do not relocate once assigned to a server.

Modeling telecommunication network problems as dynamic games has produced Nash equilibria solutions in many settings such as capacity allocation in routing [11], congestion control in product form networks [12], flow control in Markovian queuing networks [8], and combined routing and flow control [2]. Incorporating electronic market concepts such as pricing congestible network resources has been shown to encourage efficiency [15]. Network resource allocation where users were charged per unit time has also been investigated under various criteria of fairness [9; 10].

9. CONCLUSION

We presented a simple means of decentralized resource allocation that prioritizes agents' computation through a single parameter, endowment size. To utilize the mechanism, we derived a bidding policy that minimizes an agent's execution time for an itinerary, while preserving a fixed budget constraint. Furthermore, we proved that the use of our optimal bidding strategy results in a computable unique Nash equilibrium. We simulated a network of mobile agents and their hosts using our bidding strategy and resource-allocation policy to show that agents are prioritized by their endowments, our system adapts to handle situations when requests dominate capacity, and that agents' performance degrades gracefully as network delays increase.

APPENDIX

Proof of Concavity of $g_i(\theta_1)$

For the variables $\alpha^i, \beta^i, \gamma^i$ defined in equations (12)-(14) we drop the superscripts and consider the bidding function of only a single agent. Let $w(x)$ be a function defined as:

$$w(x) = -x^2 + \sqrt{x^4 - bx^3 + b\alpha x^2} \tag{29}$$

where $x \in (0, \alpha)$ and $b = (4\gamma^2/\beta)$.

Then, $g_1(\theta_1) = (1/2\gamma^2)w(\alpha - \beta\theta_1)$ for $\theta_1 \in O_1$ and

$$\partial^2 g_1/\partial\theta_1^2 = (\beta^2/2\gamma^2)(\partial^2 w/\partial x^2). \tag{30}$$

To prove the concavity of g_1 on O_1, it suffices to show that $(\partial^2 w/\partial x^2) < 0$ for $x \in (0, \alpha)$. We have

$$\partial^2 w/\partial x^2 = (2p(x)p''(x) - p'(x)^2 - 8p(x)^{\frac{3}{2}}/4p(x)^{\frac{3}{2}}), \tag{31}$$

where $p(x) = x^4 - bx^3 + b\alpha x^2$. Since $p(x) > 0$ for $x \in (0, \alpha)$, it is sufficient to show that

$$v(x) = 2p(x)p''(x) - p'(x)^2 - 8p(x)^{\frac{3}{2}} < 0. \tag{32}$$

After substituting for $p(x)$ and simplifying, we get

$$\begin{aligned} v(b, x) = \quad & -4\alpha b^2 x^3 + 12\alpha b x^4 + 3b^2 x^4 + 8x^6 \\ & -12bx^5 - 8(x^4 - bx^3 + b\alpha x^2)^{\frac{3}{2}} \end{aligned} \tag{33}$$

We note that $v(0, x) = 0 \;\; \forall x$. Taking the partial derivative of $v(x)$ w.r.t. b, we get

$$\frac{\partial v}{\partial b} = -(\alpha - x)[6bx^3 + 12x^2\gamma(x)] - 2b\alpha x^3 \tag{34}$$

$$\gamma(x) = \sqrt{x^4 + bx^2(\alpha - x)} - x^2. \tag{35}$$

Hence $\partial v/\partial b$ is negative for $x \in (0, \alpha)$ and $b > 0$. Therefore, we have $v(b, x) < 0$ for all $b > 0$ for $x \in (0, \alpha)$, which implies $(\partial^2 w/\partial x^2) < 0$ for $x \in (0, \alpha)$, and thus $(\partial^2 g_1/\partial\theta_1^2) < 0$ on O_1.

Acknowledgments

This work is supported in part by Department of Defense contract MURI F49620-97-1-0382, DARPA contract F30602-98-2-0107 and an ARO/EPRI contract.

A. REFERENCES

[1] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network Flows: Theory, Algorithms, and Applications.* Prentice Hall, Upper Saddle River, NJ, 1993.

[2] E. Altman, T. Başar, and R. Srikant. Nash equilibria for combined flow control and routing in networks: Asymptotic behavior for a large number of users. In *Proceedings of 38th IEEE Conference on Decision and Control*, Phoenix, AZ, Dec. 1999.

[3] T. Başar and G. J. Olsder. *Dynamic Noncooperative Game Theory.* Classics in Applied Mathematics, SIAM, 1999.

[4] J. Bredin, D. Kotz, and D. Rus. Market-based resource control for mobile agents. In *Proceedings of the Second International Conference on Autonomous Agents*, pages 197–204, Minneapolis, MN, May 1998. ACM Press.

[5] J. Bredin, D. Kotz, and D. Rus. Utility driven mobile-agent scheduling. Technical Report PCS-TR98-331, Dartmouth College, May 1998.

[6] J. Bredin, D. Kotz, and D. Rus. Mobile-agent planning in a market-oriented environment. Technical Report PCS-TR99-345, Dartmouth College, May 1999.

[7] K. Calvert and E. Zegura. GT-ITM: Georgia Tech internetwork topology models, 1996. `http://www.cc.gatech.edu/fac/Ellen.Zegura/gt-itm/gt-itm.tar.gz`.

[8] M. T. Hsiao and A. A. Lazar. Optimal decentralized flow control of Markovian queueing networks with multiple controllers. *Performance Evaluation*, 13:181–204, Nov. 1991.

[9] F. P. Kelly. Charging and rate control for elastic traffic. *European Transactions on Telecommunications*, 8:33–37, 1997.

[10] F. P. Kelly, A. K. Maulloo, and D. K. H. Tan. Rate control in communication networks: shadow prices, proportional fairness and stability. *Journal of the Operational Research Society*, 49:237–252, 1998.

[11] Y. Korilis and A. Lazar. Architecting noncooperative networks. *IEEE Journal on Selected Areas in Communications*, 13:1241–1251, Sept. 1995.

[12] Y. Korilis and A. Lazar. On the existence of equilibria in noncooperative optimal flow control. *Journal of the Association for Computing Machinery*, 42:584–613, May 1995.

[13] D. B. Lange and M. Oshima. Seven good reasons for mobile agents. *Communications of the ACM*, 42(3):88–89, Mar. 1999.

[14] C. Langton, R. Burkhart, M. Daniels, and A. Lancaster. The Swarm simulation system, 1999. `http://www.santafe.edu/projects/swarm`.

[15] J. K. Mackie-Mason and H. R. Varian. Pricing congestible network resources. *IEEE Journal of Selected Areas in Communications*, 13(7):1141–1149, Sept. 1995.

[16] R. T. Maheswaran, Ç. Imer, and T. Başar. Agent mobility under price incentives. In *Proceedings of 38th IEEE Conference on Decision and Control*, Phoenix, AZ, Dec. 1999.

[17] O. Regev and N. Nisan. The POPCORN market— an online market for computational resources. In *Proceedings of the First International Conference on Information and Computation Economies*, pages 148–157, Charleston, SC, Oct. 1998. ACM Press.

[18] C. F. Tschudin. Open resource allocation for mobile code. In *Proceedings of The First Workshop on Mobile Agents*, pages 186–197, Berlin, April 1997.

[19] J. E. White. Telescript technology: Mobile agents. General Magic White Paper, 1996.