

A dynamic theory of incentives in multi-agent systems (Preliminary report)

Yoav Shoham*

Stanford University
Computer Science Department
Stanford, CA 94305
U. S. A.

Katsumi Tanaka†

Toshiba Corporation
Kansai Research Laboratories
8-6-26 Motoyama-minami-cho
Higashinada-ku, Kobe 658
Japan

Abstract

Motivated by problems that plague the acceptance of groupware products, as well as the challenge of designing effective multi-agent systems, we investigate the role of incentives in “public goods” settings, that is, settings in which the value of the system is supplied by its members. In such settings one encounters a prisoners-dilemma-like problem, in which each individual has a disincentive to contribute, but much benefit from the contributions of others. The natural result is that, unless special care is taken to engineer correct incentives, no contribution is made by anyone and the system has no value. Drawing on the literature on mechanism design and public goods in economics, we present a model in which to reason about such settings. Our model has several novel features, the most important of which is its dynamic component; the model describes how the value of the system is determined over time as a result of its use. We provide several results about the model, including the identification of a *critical mass* point, which is a value of the system that, if reached, is guaranteed to henceforth enable the system to engineer incentives that lead to contribution to (and thus added value of) the system.

1 Introduction

A fundamental tenet of rational behavior, and of much work in AI, is that agents are incentive driven. One way in which this assumption manifests itself in AI is in many attempts to endow programs with various notions of motivation – preferences, utilities, goals, and so on – as well as with a general mechanism (such as a planner) that acts in accordance with these motivations, the idea being that once this general mechanism

is in place, one need only specify the motivational component in order to elicit the desired behavior. One can also turn the picture on its head, and ask how the view of *people* as rational beings should impact the design of effective programs, especially those that interact with their users. Of course, this too has been an important strand of AI research, perhaps best exemplified by application of plan- and goal-recognition techniques to question-answering systems (for example, Cohen *et al.*'s [Cohen *et al.*, 1982]).

Reasoning about incentives in a multi-agent setting is complicated, since the incentives offered to an agent may depend not only on its behavior but also on the behavior of others. The area of *mechanism design* [Fudenberg and Tirole, 1991] (also known as *implementation theory*) in economics studies the crafting of incentive structures that lead to desired behaviors. There has already been work in AI that embraces some ideas from mechanism design [Zlotkin and Rosenschein, 1993; Shoham and Tennenholtz, 1997]. Here we continue in that tradition, but concentrate on an important particular case, which in the economics jargon is referred to as *public goods*.

The prototypical example of public goods are tax-financed highway construction or national defense. In such situations there is some “public good” that is paid for by some segment of the population, but that can be enjoyed equally by everyone regardless of actual payment made (see caveat below). It is easy to see that in the absence of (e.g., judicial) external incentives, there is no incentive for any individual to pay taxes, but a great benefit from having other people pay *their* taxes. Turning to computational applications that motivate our work, consider a shared database, one that is populated by items contributed by the set of users themselves (these have been called *discretionary databases* [Connolly and Porter, 1990]). The various users (and contributors) might be people or software agents. In either case, each user derives benefit from having a rich database to draw from, but has a slight disincentive to actually contribute anything. The result is that the database remains empty, to the detriment of all users; this is of course a manifestation of a multi-person version of the prisoners dilemma.

Lest this sound too academic, let us mention that

*This work was supported by NSF grants IRI-9220645 and IRI-9503109.

†This work was carried out while the author was a visiting scholar at Stanford University.

the problem is very real. In the case of groupware it is well recognized, as documented for example in [Markus and Connolly, 1990]. We experienced it ourselves in our experiments with Fab, a collaborative recommendation system [Balabanovic and Shoham, 1997]. In addition, while it is still early days for software agents, we conjecture that the problem will be real there as well. Indeed, several researchers share the belief that future software agents will exhibit strategic interactions similar to those of people (cf. [Varian, 1995; Zlotkin and Rosenschein, 1993]).

The basic question we will ask is whether the incentives can be meddled with so that the users are incented to contribute. Stated this generally, this is the problem tackled by public-good theory in economics. However, as we shall see, our computational orientation will cause us to diverge from that literature. One divergence is the degree of freedom we allow in meddling with the incentives. Public-goods theory includes the notion of *exclusion*, which refers to the ability to withhold the benefits of the public good from some people. Given our intended applications, we will allow greater freedom. We will inherit from the public-good model the assumption that the incentives are derived entirely from the value of the system (e.g., the database), but will allow the system to offer each user any amount of benefit between zero and the entire value of the system, based on their contribution.

We inherit from the economic literature the assumption that, given a particular incentive structure, the users will contribute in a way that maximizes their rewards. But here is where we further diverge from classical work in mechanism design and public-good theory; we add a function that determines how the value of the database evolves as a result of the various contributions to it. This generates a dynamic model, which is absent from the model one finds in economics: At each stage the current value of the database determines the incentive structure, the incentive structure determines the various contributions, the various contributions together with the current value of the database determine its next value, and the process repeats.

Once we have this model in place, we can meaningfully ask various questions, such as: Does the current value of the database allow an incentive structure that incents at least one person to contribute? Is there some minimum value for the database (a “critical mass”) above which we are guaranteed that the database value will grow monotonically in time? As it turns out, these are two of the questions we will answer once we have presented the formal model.

The paper is structured as follows. In section 2 we present the formal model, accompanied by intuitive explanations and some algebraic examples. In sections 3 and 4 we present our results that were derived within the model, concerning the existence of appropriate reward schemes that lead to continuing participation in the system; from the technical point of view, these sections contains the main “results” of the paper. We conclude

in section 5 with various comments, discussion of related work, and our own planned future work in this area.

2 A dynamic model of public goods

In the following definitions we assume some fixed number n of *agents*, a set $E = [0..Ω] \subset \mathcal{R}$ of *possible effort levels*, and a set $\Theta = [0, \infty)$ of user types. In the following, $\mathcal{R}$ denotes the real numbers and $\mathcal{R}^+$ the non-negative reals.

We first define how the value of the system (e.g., database) evolves.

Definition 1 A (group-based) system growth function is a function $f : E^n \times \mathcal{R}^+ \rightarrow \mathcal{R}^+$ defined by

$$f((e_1, \dots, e_n), W) = w_1(\sum_{i=1}^n e_i) + w_2(W)$$

where w_1 and w_2 are continuous and monotonic increasing functions.

Intuition: The system’s value is a function of its old value and the incremental efforts put into it by each of the n agents. Note that the growth depends only on the sum on the efforts; in particular, it does not depend on the identities of the contributors. Also note that ‘growth’ can be negative.

This defines one direction of the dynamics, namely how the value of the system is determined by the agents. We now define the more involved way in which the system provides incentives to the agents and thus, indirectly, leads them to behave in certain ways.

Definition 2 A (system-based) reward function is a function $r : E \times E^n \times \mathcal{R}^+ \rightarrow \mathcal{R}^+$ and must satisfy $r(e, (e_1, \dots, e_i, \dots, e_j, \dots, e_n), W) = r(e, (e_1, \dots, e_j, \dots, e_i, \dots, e_n), W)$ for all $i, j \in \{1, \dots, n\}$

Intuition: Based on its own most recent effort level e , efforts of all agents $(e_1, \dots, e_n)$, and on the current value W of the system (e.g., the database), the agent is given a certain reward $r(e, (e_1, \dots, e_n), W)$. Note that rewards are symmetric; they can not favor one agent over another based on any consideration other than the effort made by the agents.

Definition 3 A disutility function is a function $d : E \times \Theta \rightarrow \mathcal{R}^+$, such that

$$\text{for all } \theta, d(\Omega, \theta) = \infty,$$

$$\text{for all } \theta, d(0, \theta) = 0,$$

$d(e, \theta)$ is a continuous and twice differential function of e and θ ,

$$\frac{\partial d}{\partial e} > 0, \frac{\partial^2 d}{\partial e^2} > 0, \frac{\partial d}{\partial \theta} < 0, \frac{\partial^2 d}{\partial \theta^2} < 0$$

Intuition: Each agent experience a certain disutility from engaging in effort; the larger the effort, the greater the disutility. The disutility function is concave; disutility increases more rapidly the higher the effort, and is infinite at the level of maximum effort (one concrete interpretation is to view effort level as hours of work in a day, and Ω as 24). However, different agents experience different disutilities at different levels (again,

for concreteness, think of different sleep requirements of people). This is reflected in the agent type; the higher the type, the lower is the disutility at any given effort level.

A reward function and a disutility function form the basis for a composite utility function:

Definition 4 A (system-based) utility function is a function $u : E \times E^n \times \mathcal{R}^+ \times \Theta \rightarrow \mathcal{R}$ defined by

$$u(e, (e_1, \dots, e_n), W, \theta) = r(e, (e_1, \dots, e_n), W) - d(e, \theta)$$

where r is a (system-based) reward function, and d is a disutility function.

Note that for any W, θ and $e_1, \dots, e_n$ other than e , we have $u(0, (e_1, \dots, e_n), W, \theta) \geq 0$. $u(0, (e_1, \dots, e_n), W, \theta)$ is called the *reservation value* of the agent; in order to get the agent to take any action at all the system must generate at least this great an incentive.

Let us now tie this all into a definition of a dynamic public-good system:

Definition 5 A dynamic public-good system (DPGS) is a tuple $((\theta_1, \dots, \theta_n), r, d, f)$, where $\theta_i \in \Theta$, r is a reward function, d is a disutility function, and f is a growth function.

Intuition: We have n agents, where θ_i is the type of the i th agent. They are all subject to the same reward and disutility functions, though, of course, they will experience different values due to their different types and the different efforts they will make.

Definition 6 A DPGS $((\theta_1, \dots, \theta_n), r, d, f)$ with $f = w_1(\sum_{i=1}^n e_i) + w_2(W)$ is said to be *endogenous* if $r(e, (e_1, \dots, e_n), W)$ is constrained by $f = w_1(\sum_{i=1}^n e_i) + w_2(W)$.

Intuition: In endogenous systems all the rewards come from within the system. However, agents can be “promised” a reward based on the future value of the system (e.g., people are persuaded to pay for a highway upgrade based on the value of the upgraded highway, not the existing one). This is crucial if we wish for the system value to increase with time.

In the sequel we will be concerned mostly with endogenous systems.

We are close to having the dynamics of the system completely specified. Clearly, given an initial value of the system, if we knew what effort levels would be selected by the each agent for each value of the system we would be able to predict a deterministic evolution of the system’s value.

But rather than take these effort levels as given, we try to deduce them under the assumption that all the agents are economically rational. Specifically, we assume that agents select effort levels that maximize their utilities. The problem, of course, is that this is not well defined in general, since each agent’s reward (and thus utility) depends in part on the efforts made by all the agents. Indeed, this situation defines a game in the economic sense, and we can apply the various solution concepts from game theory [Osborne and Rubinstein, 1994].

However, we are really not looking to solve a particular game, but are rather asking the inverse question, namely how can we craft a reward function that induces a game with desired properties. This is precisely the realm of mechanism design [Fudenberg and Tirole, 1991].

Traditional work on mechanism design assumes direct manipulation of the utilities. We have only roundabout (and, because of the restriction to endogenous systems, constrained) ways of doing so, via manipulation of the reward function. There are potentially many sorts of mechanisms we can attempt to set up here, but we will be looking at a very particular case, one based on *dominant strategies*.

Definition 7 A vector of efforts $(e_1^*, \dots, e_n^*)$ is said to be *dominant* for a DPGS $((\theta_1, \dots, \theta_n), r, d, f)$ at W iff for all $i \in \{1, \dots, n\}$, and all $e_1, \dots, e_n \in E$, it is the case that $u(e_i^*, (e_1, \dots, e_i^*, \dots, e_n), W, \theta_i) \geq u(e_i, (e_1, \dots, e_i, \dots, e_n), W, \theta_i)$

Intuition: This is a strong requirement. It says that each agent can choose an effort level that is guaranteed to maximize his utility, no matter what the other agents do. If such an effort level exists, surely we can assume that the agent will adopt it. Therefore, if we are guaranteed that a given DPGS has a dominant effort vector for *any* value of the system, we can predict a deterministic evolution of the system value.

In section 3 we will look at conditions in which we can devise reward functions that lead to dominant strategies. Of course, the fact that a strategy is dominant doesn’t imply that it is a good strategy, and in particular that it leads to growth in the value of the system; we will be looking for dominant strategies that do have this property.

We conclude the section with an example that illustrates the definitions.

Example 1 Consider a community-based news clipping service, implemented as a discretionary database. During the day each user can annotate news articles s/he came across with some keywords to facilitate indexing, and place it in the database. At the end of the day each user can query the database; for simplicity, assume that each query returns a single item. We can model the situation with the following DPGS:

$e_i \in [0.. \Omega]$: the number of items placed in the database by user i during the day. Assume for simplicity that no two users submit the same article.

W : the number of items in the database at the end of the day.

f : the growth function which determined the value W of the database at the end of the day, based on its value W yesterday and the items placed during the day: $f((e_1, \dots, e_n), W) = \sum_{i=1}^n e_i + 0.5 \cdot W$. f reflects an assumption that each day half the database items left over from the previous day become obsolete and are removed from the database automatically.

r : the number of articles the user i is allowed to draw from database at the end of the day (assume that

beyond that number queries result in an “account overdrawn” message): $r(e_i, (e_1, \dots, e_n), W) = f((e_1, \dots, e_n), W)$. r reflects the fact that each user is given free access to the database regardless of contribution.

d : the disutility function $d(e, \theta) = \exp(\frac{e}{(\Omega - e)\theta}) - 1$.

For simplicity, let us assume that there are two users with types $\theta_1 = 1$ and $\theta_2 = 0.5$.

Since $r(e, (e_1, e_2), W) = e_1 + e_2 + 0.5 \cdot W$, the utility function for the first user becomes $u(e_1, (e_1, e_2), W, \theta_1) = e_1 + e_2 + 0.5 \cdot W - \exp(\frac{e_1}{(\Omega - e_1)}) + 1$, and similarly for second user. The derivative of our particular utility function for either agent is therefore $u'(e) = 1 - \frac{\exp(\frac{e}{(\Omega - e)})\Omega}{(\Omega - e)^2}$, for any value of the effort level of the other agent. As for any legal utility function, since our utility function is convex it is maximized where the derivative is zero if such a point exists, or at 0 otherwise. Figure 1 plots the utility function for two values of Ω ; for $\Omega = 3$ the utility has a maximum at $e^* > 0$, and for $\Omega = 1$ the derivative is always negative and the utility is maximum at 0. The second is an extreme case of lack of incentives; under no circumstance could a user be incented to place items in the database. The first case does incent the users to exert some effort, but whether this effort is sufficient to sustain growth depends on the value of the database; we return to this issue in the next sections.

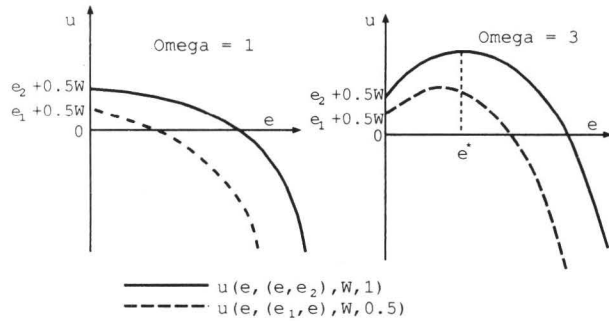


Figure 1: Utility functions

3 Reward design

As shown in example 1, careless reward schemes can lead to a prisoners-dilemma situation, in which no agent contributes to the system, and the system dies. We begin to investigate now the design of reward functions that avoid this situation. In this section we look at what it takes to design a value-increasing reward function at a particular value of the system. In the next section we'll look at conditions that guarantee that once such a reward can be offered, the dynamics of the system will ensure that henceforth this will be the case.

Our first theorem gives necessary and sufficient conditions on effort vectors such that the vectors are dominant strategies for some reward function. (All the proofs are omitted from this article.)

Theorem 1 *Given:*

- a *DPGS* $((\theta_1, \dots, \theta_n), r, d, f)$,
- a system value W , and
- a vector $(e_1^*, \dots, e_n^*)$ such that $0 \leq e_i^* < \Omega$,

the following two conditions are equivalent:

1. There is a reward function r such that $(e_1^*, \dots, e_n^*)$ is a dominant strategy for *DPGS* $((\theta_1, \dots, \theta_n), r, d, f)$ at W .
2. $e_i^* \geq e_j^* \equiv \theta_i \geq \theta_j$ for all $i, j \in \{1, \dots, n\}$.

Intuitively, Theorem 1 states that it is possible to design a reward function that induces any desired effort levels on the part of the various agents, so long as the effort assigned to low-disutility agents is higher than that of high-disutility agents. Note that Theorem 1 does not restrict the reward function. It is perhaps not surprising that agents can be incented to do almost anything given arbitrary incentives. The following theorem looks at the effects of endogeny:

Theorem 2 *Under the same conditions as in Theorem 1, and given an endogenous DPGS $D = ((\theta_1, \dots, \theta_n), r, d, f)$, the following two conditions are equivalent:*

1. There is a reward function r such that $(e_1^*, \dots, e_n^*)$ is a dominant strategy for D at W .
2. $-f((0, \dots, e_i, \dots, 0), W) \leq d(e_i^*, \theta_i) - d(e_i, \theta_i) \leq f((0, \dots, e_i^*, \dots, 0), W)$ for all $i \in \{1, \dots, n\}, e_1, \dots, e_n$.

Theorem 2 states the strict relationship between system growth function and disutilities of agents for endogenous *DPGS*, whose intuition is that the disutility incurred by an agent at any given effort level should be compensated by system's value. Overall these two theorems tell us that as system designers we have quite a lot of power, even when working within endogenous *DPGS*. This is illustrated in the following example.

Example 2 Let us modify Example 1 (the case of $\Omega = 1$) by adjusting the reward function r such that $r(e, (e, e'), W) = r(e, (e', e), W) =$

$$\begin{cases} \beta + 0.5 \cdot W & (e = \alpha, e' = \beta) \\ \alpha & (e = \beta, e' = \alpha) \\ \alpha + \exp(\frac{\alpha}{1-\alpha}) - \exp(\frac{\beta}{1-\beta}) & (e = \alpha, e' = \alpha) \\ \beta + 0.5 \cdot W - \exp(\frac{2\alpha}{1-\alpha}) + \exp(\frac{2\beta}{1-\beta}) & (e = \beta, e' = \beta) \\ \alpha + 0.5 \cdot W & (e = \alpha, e' \neq \alpha, \beta) \\ \alpha + 0.5 \cdot W - \exp(\frac{2\alpha}{1-\alpha}) + \exp(\frac{2\beta}{1-\beta}) & (e = \beta, e' \neq \alpha, \beta) \\ \text{Min}(\exp(\frac{e}{1-e}) - \exp(\frac{\alpha}{1-\alpha}), \exp(\frac{2e}{1-e}) - \exp(\frac{2\beta}{1-\beta})) & (\text{otherwise}) \end{cases}$$

If $\alpha \geq \beta$ there is a dominant strategy (α, β) since the following conditions are satisfied:

$$u(\alpha, (\alpha, e_2), W) \geq u(e_1, (e_1, e_2), W) \text{ for all } e_1, e_2 \in [0, \Omega) \\ u(\beta, (e_1, \beta), W) \geq u(e_2, (e_1, e_2), W) \text{ for all } e_1, e_2 \in [0, \Omega).$$

In order to ensure that this *DPGS* is endogenous, r must satisfy $0 \leq r(e, (e, e'), W) \leq f((e, e'), W)$ for all e ,

e' , which corresponds to the following conditions:
 $-e - 0.5 \cdot W \leq \exp(\frac{\alpha}{1-\alpha}) - \exp(\frac{e}{1-e}) \leq \alpha + 0.5 \cdot W$,
 $-e - 0.5 \cdot W \leq \exp(\frac{2\beta}{1-\beta}) - \exp(\frac{2e}{1-e}) \leq \beta + 0.5 \cdot W$
for all e .

4 Critical mass

The previous section discussed our ability to engineer good incentives at a particular system value. We now take this a step further and ask whether we can have some assurance that from a certain point on system will continue to be used and to grow monotonically in value.

The key notion here is that of *critical mass*:

Definition 8 The critical mass of a DPGS $((\theta_1, \dots, \theta_n), r, d, f)$ is the smallest system value W_c such that

$W_c \in \mathcal{W} =_{\text{def}} \{W \in \mathcal{R}^+ : f((\Omega, \dots, \Omega), W) > W\}$,
and

for all $W \in \mathcal{W}, W \geq W_c$ there exists some function r' such that

1. $((\theta_1, \dots, \theta_n), r', d, f)$ is endogenous.
2. there exists a strategy $(e_1^*, \dots, e_n^*)$ that is dominant for $((\theta_1, \dots, \theta_n), r', d, f)$ at W , and such that $f((e_1^*, \dots, e_n^*), W) > W$.

The intuition is as follows. Clearly, system growth increases monotonically with the total effort made, and so positive growth is achieved by some combination of efforts if and only if it is achieved by maximal efforts by all agents. By definition, above the critical mass it must be the case that at all system values in which positive system growth is possible at all, agents can be endogenously induced to in fact exert sufficient effort to enable positive (though not necessarily maximal) growth.

Note that this definition could be stronger; it requires an appropriate reward function at every $W \in \mathcal{W}, W \geq W_c$, but these functions need not be the same. Furthermore, we give no guidance about how to compute these reward functions.

The critical mass does not always exist; here is a necessary and sufficient condition for its existence.

Theorem 3 Let the growth function of a DPGS D be $w_1(\Sigma e_i) + w_2(W)$ as in Definition 1. Then D has a critical mass W_c if and only if there exists a W such that

there exists a effort level $\bar{e}$ such that

$$\begin{aligned} 0 &\leq \bar{e} < \Omega, \\ w_1(\bar{e} \cdot n) + w_2(W) &> W, \text{ and} \\ d(\bar{e}, \theta_i) &\leq w_1(\bar{e}) + w_2(W) \text{ for all } i \in \{1, \dots, n\} \end{aligned}$$

Intuitively, this says that there exists a critical mass iff there would be enough compensation for every agent's disutility in the case that maximum effort of each agent is exerted. Unfortunately, we provide no guidance here about how to compute this critical mass, or even how to determine whether the condition stated holds in any given circumstance.

However, we do end with one bit of good news. One might worry that if despite the meager instructions offered here one found a critical mass, the increments of the system value might not promised. The following theorem guarantees that this is not the case; once the critical mass is reached, the system's value will increase.

Theorem 4 Given an endogenous DPGS D , a system value W , and a dominant strategy $(e_1^*, \dots, e_n^*)$ for D at W , then for every W' there exists an endogenous DPGS D' such that

1. $D = D'$ except possibly in the reward function,
2. there exists a dominant strategy $(e_1^{*'}, \dots, e_n^{*'})$ for D' and W' , and
3. $\sum_{i=1}^n e_i^{*'} > \sum_{i=1}^n e_i^*$ if $W' > W$.

Recall that the growth function is a function of the current system value and the sum of the efforts of the agents. Therefore beyond the critical mass the system value will increase monotonically within $\mathcal{W}$.

We illustrate these definition and properties in the following example.

Example 3 Let us modify Example 2 to realize system growth. Suppose there exists W_c which is the smallest value such that all $W < 4$ (corresponding to W) and $W \geq W_c$ satisfy the following conditions for all $\alpha < \bar{e}, \beta < \bar{e}$ such that $\alpha \geq \beta, \bar{e} < 1$.

$-e - 0.5 \cdot W \leq \exp(\frac{\alpha}{1-\alpha}) - \exp(\frac{e}{1-e}) \leq \alpha + 0.5 \cdot W$, and
 $-e - 0.5 \cdot W \leq \exp(\frac{2\beta}{1-\beta}) - \exp(\frac{2e}{1-e}) \leq \beta + 0.5 \cdot W$ for all $e \in [0, \bar{e}]$.

Then clearly W_c is a critical mass point. To see why, note that we can design a reward function which gives a dominant strategy (α, β) for an endogenous DPGS at W such that the next system value W' will be $\alpha + \beta + W \cdot 0.5 > W$, because $W < 4 \cdot \bar{e}$ and α, β can be arbitrarily set in the range $0 \leq \alpha \leq \bar{e}, 0 \leq \beta \leq \bar{e}$. Furthermore, Theorem 4 says that there exists a dominant strategy (α', β') at W' such that $\alpha' + \beta' > \alpha + \beta$. Obviously $\alpha' + \beta' + W' \cdot 0.5 > \alpha + \beta + W \cdot 0.5$ holds, which promises monotonic increasing system growth in the future.

5 Final remarks

We have provided a formal model of a multi-agent public-goods system. Given that our model draws heavily on work in economics, it is important that we identify the particular contributions of this paper. These contributions include the following:

- Allowing arbitrary reward functions that don't exceed the system's value.
- Defining a dynamic model in which agents' efforts and system value are mutually reinforcing.
- Providing various results about this model, which (to our knowledge) have no analogue in existing multi-agent models. These results include necessary and sufficient conditions for the existence of a dominant strategy.

- Defining in the model the notion of a critical mass, the point beyond which the continued well being of the system is made possible, and providing necessary and sufficient (albeit non-constructive) conditions for the existence of the critical mass.

Our model is based on notions from micro-economics, but also has several novel features. One is the total freedom to choose reward functions within the current system value. A second, more fundamental one is the construction of a dynamic model.

While the literature on public goods, agency theory, and mechanism design is too rich for us to cover here, the literature on applications of mechanism design and/or public goods in AI or software design is very limited. One notable exception is [Avery *et al.*, 1996]’s discussion of a pricing method for inducing optimal provision of evaluations (as in restaurant reviews, or Web-page ratings) from users. They regard the database of evaluations as a public good, and discuss implementability of payment scheme between evaluators and readers. They too consider a dynamic model, but unlike us they do not assume a variable effort but a fixed cost. [Varian, 1995]’s discussion of the application of agency theory to software design is related insofar as it too examines the application of agency theory – the asymmetric, two-person case of mechanism design – in the software domain. However, the similarity ends there, as he does not consider the public good aspect at all, let alone its dynamic version. Other work in generally in the area of application of economic mechanisms in AI includes Market Oriented Programming [Wellman, 1996], [Zlotkin and Rosenschein, 1993], and [Shoham and Tennenholtz, 1997]; however, there is no direct connection between the technical development nor the motivation here and any of these three.

Several challenges remain, in the areas of models, algorithms, and application. Our model makes several assumptions, which we mention throughout the construction. First, we have looked only at dominant pure strategies. It is natural to relax this strong requirement and look at general Nash equilibria among possibly mixed strategies. Second, we have assumed a particular additive form of the system’s growth function, the result being that the system’s value can grow unboundedly. This is unrealistic in any interesting domain; for example, in a discretionary database it is unrealistic to assume that no matter how many people post items to the database, these items are always distinct. It would be natural to allow more complex growth functions which model the phenomenon of diminishing returns. Moving beyond the limitations of the model itself, we note that we have given no algorithms. In particular, we have given no algorithm to compute the critical mass, nor to determine whether it exists. Finally, as was mentioned in the introduction, this work was motivated by real difficulties encountered in incenting users to participate in groupware. We hope to incorporate the lessons learned from this work in future groupware design.

Acknowledgments. The authors are grateful for discussion with members of the Nobots group and for com-

ments by anonymous reviewers.

References

- [Avery *et al.*, 1996] Chris Avery, Paul Resnick, and Richard Zeckhauser. The market for evaluations. In Harvard University Mimeo, <http://www.research.att.com/~presnick/papers/mfe/>, 1996.
- [Balabanovic and Shoham, 1997] Marko Balabanovic and Yoav Shoham. Combining content-based and collaborative recommendation. In *Communications of the ACM, Special Issue on Recommender Systems (to appear)*, 1997.
- [Cohen *et al.*, 1982] P. R. Cohen, C. R. Perrault, and J. Allen. Beyond question-answering. In W. Lehnert and M. Ringle, editors, *Strategies for natural language processing*. L. Erlbaum Associates, 1982.
- [Connolly and Porter, 1990] Terry Connolly and Alan. L. Porter. Discretionary databases in forecasting. *Journal of Forecasting*, 9:1–12, 1990.
- [Fudenberg and Tirole, 1991] Drew Fudenberg and Jean Tirole. Bayesian games and mechanism design. In *Game Theory*. MIT Press, 1991.
- [Markus and Connolly, 1990] M. Lynne Markus and Terry Connolly. Why CSCW applications fail: Problems in the adoption of interdependent work tools. In *CSCW ’90*, pages 371–380, 1990.
- [Osborne and Rubinstein, 1994] M. J. Osborne and A. Rubinstein. *Game Theory*. MIT Press, 1994.
- [Shoham and Tennenholtz, 1997] Y. Shoham and M. Tennenholtz. Emergent conventions in a game theoretic setting. *AI Journal special issue on Economic Foundations of Multi-Agent Systems (to appear)*, 1997.
- [Varian, 1995] Hal R. Varian. Mechanism design for computerized agents. In Usenix Workshop on Electronic Commerce, <http://www.sims.berkeley.edu/~hal/people/hal/papers.html>, July 1995.
- [Wellman, 1996] Michael P. Wellman. Market-oriented programming: Some early lessons. In S. Clearwater, editor, *Market-Based Control: A Paradigm for Distributed Resource Allocation*. World Scientific, 1996.
- [Zlotkin and Rosenschein, 1993] Gilad Zlotkin and Jeffrey S. Rosenschein. A domain theory for task oriented negotiation. In *Proc. of IJCAI-93*, pages 416–422, 1993.