Work Accounting Mechanisms:
Theory and Practice/

Sven Seuken_y, Michel Meulp older_z, Dick H. J. Ep ema_z,
David C. Parkes_y, Johan A. Pouwelse_z, Jie Tang_x

Decemb er 15, 2010

PRELIMINARY DRAFT!

Abstract The Internet has enabled a new paradigm of economic pro duction, where individual users p er-form work for others, often in small units, for short p erio ds of time, and without formal contracts or monetary payments. These distributed work systems can arise in many places, for example in p eer-to-p eer (P2P) flle-sharing networks, in ad-ho c wireless routing networks, or in 3G content distribution systems. The particular challenge is to incentivize users to p erform work for others, even though all interactions are bilateral and monitoring is not p ossible. In this pap er, we formalize the problem of designing incentive-compatible work accounting mechanisms that measure the net contributions of users, despite relying on voluntary rep orts. We describ e a fully distributed accounting mechanism called BarterCast and intro duce the Drop-Edge extension which removes any incentive for a user to make misrep orts ab out its own interactions. We prove that the information loss necessary to achieve this incentive compatibility is small and vanishes in the limit as the numb er of users grows. In some domains, users may b e able to cheaply create fake identities (i.e, sybils) and use those to manipulate the accounting mechanism. A striking negative result is that no sybil-pro of accounting mechanism exists if one requires resp onsiveness to a single p ositive rep ort. To evaluate the welfare prop erties of BarterCast+DropEdge, we flrst present results from a discrete, round-based simulation, showing that the mechanism achieves very high e–ciency. We have also implemented the mechanism in Tribler , a BitTorrent software client, that is already deployed in the real world and has thousands of users. Exp erimental results using Tribler demonstrate that the mechanism successfully prevents free-riding in P2P-flle-sharing systems, and achieves b etter e–ciency than the standard BitTorrent proto col.

Keywords: Mechanism Design, Distributed Systems, Accounting.

1 Introduction

Distributed work systems arise in many places, for example in p eer-to-p eer (P2P) flle-sharing networks, where users share videos, music or software with each other, or in ad-ho c wireless networks where individual p eers route data packages for each other through the network. Amazon Mechanical Turk and other \crowd sourcing" applications suggest an explosion of interest in this distributed form of pro duction. Within AI, distributed work systems contribute to the agenda on multi-agent resource allo cation, enhancing our understanding of architectures to co ordinate artiflcial agents.

Of course, the total amount of work p erformed by a p opulation must equal the total amount of work consumed. Moreover, while some degree of free-riding may b e acceptable (e.g., if some users are altruistic while work is extremely costly for others), it is generally accepted that the long-term viability of work systems that op erate without the use of monetary transfers must rely on roughly balanced work contributions. Otherwise, a ma jority of the agents may seek to free-ride on the system, i.e., minimize the work they p erform for others and maximize the work they can consume. Current systems often enforce temp orally-lo cal balances, e.g., via fair exchange proto cols where agent A only continues to p erform work for agent B if agent B recipro cates immediately. Yet, this \lo cal balance" clearly intro duces a large ine–ciency{ users are limited to consuming work at a rate at which they can themselves pro duce work, must b e able to simultaneously consume and pro duce work, and cannot p erform work and store credits for future consumption [17]. For example, [18] found that more than 80% of BitTorrent users go o†ine immediately once they have flnished downloading. Accounting mechanisms would solve this problem by keeping long-term tallies of work p erformed and consumed by each user. This would give users an incentive to share even after they have flnished downloading, and thus increase system e–ciency.

The particular challenge that we address here o ccurs when the interactions are bilateral and there is no ability for a third party to monitor the activities. We consider distributed work systems where the agents p erform small units of work for each other, for limited p erio ds of time, such that no contract covers the interaction, b ecause the efiort for creating and legally enforcing the contracts would b e to o high relative to the value of the work. This rules out the use of any kind of real or virtual currency, b ecause the institutional requirements for the exchange of payments are not available. Furthermore, we assume that there is no a-priori trust relationship b etween the agents in the network that could b e used, but instead, that an agent can only earn trust by p erforming work. Because there is no center monitoring the interactions, the accounting mechanism must rely on voluntary rep orts by the same users that p erform and consume work in the system. Thus, an accounting mechanism must b e robust to manipulations, b ecause users might otherwise overstate the amount of work contributed, or understate the amount of work consumed. It is our goal to design accounting mechanisms that are incentive-compatible in the sense that it is in each individual user's b est interest to make truthful rep orts ab out his interactions with other users.

1.1 Accounting vs. Reputation Mechanisms

The problem of designing an incentive-compatible accounting mechanism shares some features with work on trust/reputation mechanisms [9]. However, there are signiflcant difierences which make reputation and accounting mechanisms incomparable. First, and somewhat informally, the essence of accurate reputation aggregation is the op eration of averaging whereas the essence of accurate accounting is the op eration of addition. In a reputation system like eBay, individual users provide feedback ab out each other, and the individual feedback rep orts of two difierent agents regarding a third agent could b e very difierent. The task of the reputation system is to aggregate multiple rep orts

into one overall reputation score; in a sense, \averaging" over all rep orts. In contrast, in distributed work systems, multiple rep orts ab out work consumed or p erformed by an agent simply need to b e added together, to eventually determine the overall net contributions of that agent.

Second, in distributed work systems, every p ositive rep ort by A ab out his interaction with B , i.e., B p erformed work for A, is simultaneously a negative rep ort ab out A, i.e., A received work from B . This fundamental tension is not present in reputation mechanisms where a go o d rep ort by A ab out B do es not re ect badly on A. Third, mechanisms that are sybilpro of in traditional reputation systems are not necessarily sybilpro of in distributed work systems.

1.2 Real-world applications for Accounting Mechanisms

The most-widely used application that falls into the category of distributed work systems we study is p eer-to-p eer flle-sharing. The p erformance of P2P flle sharing systems crucially dep ends on the contribution of resources by their participating users. None of the existing proto cols provides longterm sharing incentives to its users. Free-riding is a well-known issue in P2P research; its efiects have b een empirically measured [1, 21] and extensively analyzed [3, 7, 13, 15]. None of the existing systems provides the users with a long-term incentive to upload data to other users. Thus, P2P flle-sharing could b eneflt a lot from a well-designed accounting mechanism.

Another interesting application for accounting mechanisms is content distribution in 3G networks. 3G bandwidth is a very scarce resource, downloading data via a 3G network is relatively slow and requires a lot of battery p ower from a smartphone or similar device. In contrast, the supply of bandwidth is wireless networks is relatively cheap, the download sp eed in such networks is very fast, and smartphones requires much less battery p ower to connect to a wireless network than to a 3G network. Recently, researchers have started to lo ok into the p otential of using ad-ho c wireless networks to distribute certain data instead of using 3G bandwidth [4]. Imagine, you are at the train station at 9am in the morning, and lots of p eople are downloading the news, the weather, etc. onto their smartphones. Whenever multiple near-by users are downloading the same data, the e–ciency could b e increased, if only one of the users downloads the data via the 3G network, and then the data is distributed via ad-ho c wireless networks to the other users. This is already technically feasible and can happ en without any user interaction. However, given that 3G connections are costly for the user (draining the battery, and accumulating MBs that count towards the monthly download limit), with a naive implementation, no users would like to b e the one who downloads the data packages via 3G and then distributes them. By using an accounting mechanism like the one we prop ose in this pap er, we can solve this problem, balancing the work load over time, and giving each user an incentive to consume and p erform work at difierent p oints in time.

A third application concerns routing in ad-ho c networks. Imagine you need to set up a communication network in an area without existing communication infrastructure, for example, in an area that has just b een hit by a natural disaster or in a war zone. In such situations, oftentimes difierent organizational units, p otentially from difierent countries, set-up camps next to each other, but have no direct relationship with each other. Ad-ho c wireless networks are a very e–cient way of quickly establishing a communication network in these situations, however, bandwidth will generally b e scarce, and routing data packages from other organization units through your own network might reduce the amount of bandwidth you can use yourself. Again, a prop erly designed accounting mechanism can address the problem by making sure that over time, each networks routes approximately the same numb er of data packages, thereby providing the difierent network op erators with an incentive to collab orate in the set-up of these networks.

3

1.3 Outline and Overview of Results

In this pap er we present a theoretical and exp erimental analysis of accounting mechanisms for distributed work systems. In Section 2 we formalize the concept of a distributed work system and intro duce BarterCast, a fully decentralized, lightweight information exchange system. In Section 3 we present what we b elieve to b e the flrst formal mo del for the design of incentive compatible accounting mechanisms. We show how the BarterCast-basic mechanism is susceptible to misrep ort manipulations, and we intro duce the Drop-Edge extension that removes any incentives for agents to misrep ort information, by selectively dropping information dep endent on the decision context. We provide a theoretical analysis of Drop-Edge in Section 4.1 where we show that the information loss due to dropping some of the information is small and vanishes in the limit as the numb er of agents in the network gets large. In Section 4.2 we analyze the problem of sybil attacks on accounting mechanisms. We flrst show that, if we require that a single p ositive rep ort ab out an agent shall make that agent b etter ofi than a new-comer, then no sybil-pro of accounting mechanism exists. However, we show how we can design accounting mechanisms that are sybil-pro of against certain kinds of attacks when we relax this requirement. In Section 5 we provide an extensive exp erimental evaluation of the BarterCast-DropEdge mechanism in a discrete simulation environment. First, we show that the approximation ratio of Drop-Edge is very go o d in practice. Next, we show e–ciency results comparing BarterCast-basic and BarterCast-Drop-Edge. We show convincingly that strategic agents that manipulate the accounting mechanism b eneflt signiflcantly in BarterCast-basic such that the p erformance of co op erative agents sufiers signiflcantly. In contrast, BarterCast-DropEdge is robust against misrep ort attacks, and thus strategic agents are not doing any b etter than standard free-riders, and co op erative agents have a signiflcantly higher e–ciency. We also show that this efiect increases over time, as agents gather more and more information ab out each other and thus are able to discriminate b etter and b etter b etween sharers and free-riders. In Section 6 we provide results from extensive exp eriments in the BitTorrent P2P flle-sharing network. We have already implemented a series of accounting mechanisms into Tribler, a real flle-sharing client that is already deployed and b eing used by thousands of users. Using Tribler we ran simulation exp eriments on the BitTorrent proto col level, giving us much more realistic simulation results ab out the p erformance of various accounting mechanisms. In Section 6.1 we explain why the requirement of b eing backwardscompatible intro duces new di–culties for making use of accounting mechanisms in BitTorrent, and in Section 6.3.2 we discuss in detail the advantages and disadvantages of using a ranking p olicy or a banning p olicy for making work allo cation decisions in BitTorrent. We show that using the ranking p olicy, the p ower of accounting mechanisms is inherently limited in BitTorrent. However, we show that using the banning p olicy, we can achieve p erformances for sharers and free-riders that difierent signiflcantly (by more than a factor of 2), and thus a b ehavioral change can b e exp ected (where freeriders b ecome sharers after a while). In Section 6.3.5 we discuss in detail the efiects of accounting accuracy and noise in large networks and explain why we can exp ect high accuracy and low noise in real BitTorrent networks. Finally, in Section 6.4, we present exp erimental results that are trace-based, thus, using an agent-arrival pro cess that we have actually observed on a real-world BitTorrent tracker. These exp eriments verify our previous results, showing that the agents' accounting scores have a very high p ositive correlation with their true net contributions to the network. Furthermore, all exp eriments show that using the BarterCast-Drop-Edge mechanism, we can separate the p erformance of sharers and free-riders signiflcantly, and assuming some kind of b ehavioral change (i.e., free-riders b ecoming sharers over time), we can achieve a total e–ciency that is higher than with the standard BitTorrent proto col.

4

1.4 Related Work

Most of the prior work related to distributed work systems and accounting mechanisms has studied the domain of P2P flle-sharing, often with the primary goal to prevent free-riding. Gupta et al. [11] present a reputation system that is partially distributed, but relies on the authority of a single agent that stores p eer reputations. As a mechanism for multiple agents is not discussed, the presented solution in essence remains centralized. Kamvar et al. [12] present EigenTrust, an algorithm for reputation management in P2P networks that is based on distributed computing of a globally consistent trust vector for every p eer in the system. EigenTrust's theoretical prop erties are attractive. On the downside, EigenTrust relies on pre-trusted p eers for convergence and its aim for global consistency assumes a rigid network of p eers. Karma [24] is a system in which sets of bank no des keep track of the transaction balance of p eers. The approach relies on a DHT-based structure of allo cating bank no des and cryptographic mechanisms for security. BitTorrent [6] uses a tit-for-tat p olicy with short-term, direct incentive prop erties. While tit-for-tat is successful in short term transactions, the mechanism ofiers no incentives for long-term sharing of content. Lian et al. [14] identify key disadvantages of b oth EigenTrust and tit-for-tat and prop ose a hybrid mo del in b etween the two.

   Piatek et al. [17] also study decentralized accounting and flnd empirically that most users of P2P flle-sharing networks are connected via a one hop link in the connection graph. They prop ose to use well-connected intermediaries to broker information, but without providing prop er incentives to the intermediaries to b ehave truthfully. Feldman et al. [7, 8] study the challenges involved in providing robust incentives in fully decentralized P2P networks, including free-riding, whitewashing, and misrep ort attacks. They intro duce a reputation mechanism based on max ow, but they implicitly assume that all no des have a complete view of the network, which is highly unrealistic in large, dynamic communities. Moreover, the authors omit any discussion regarding how no des acquire this information. Furthermore, in contrast to our prop osal, their mechanism is not misrep ort-pro of. Interestingly, our mechanism shares some similarities with a mechanism prop osed by Alon et al. (2010), who consider voting environments where the set of candidates coincides with the set of voters. However, they only consider one-shot games, and don't consider the problem of agents earning trust over time. A recent pap er by Resnick and Sami (2009) addresses sp eciflcally the question of transitive trust, and they show novel results ab out sybilpro of trust mechanisms. In their domain they do not consider shared trust information and thus they do not have to study misrep ort attacks. An interesting, but orthogonal direction, is provided by studies of virtual currency systems [10]. There, work provision or consumption is observable and there is a trusted currency. In general, we are not aware of any practically feasible mechanism that is decentralized to the same degree as our prop osal, misrep ortpro of, tested under realistic, trace-based community conditions, and deployed in a real-world P2P network.

2 Distributed Work Systems

Consider a distributed work system of n agents each capable of doing work for each other. All work is assumed to b e quantiflable in the same units. The work p erformed by all agents is captured by a work graph:

,0denoting the number of units of work.Deflnition 1. (Work Graph) A work graph $G = (V ; E ; w)$ has vertices $V = f 1; : : : ; ng$, one for each agent, and directed edges $(i; j) 2 E$, for $i; j 2 V$, corresponding to work performed by i for j , with weight $w (i; j) 2 R$
5

$$w_i(i,j) = 10$$
$$w_j(i,j) = 5$$

$$w_i(i,j) = 10, \quad w_{ii}(j,i) = 100$$

$$w_i(i,k) = 10, \quad w_k(i,k) = 0$$
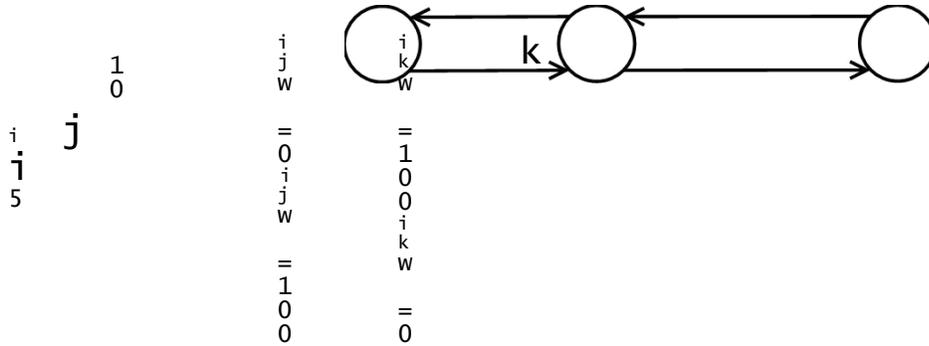$$w_{ii}(i,k) = 100$$

Figure 1: A sub jective work graph from agent i's p ersp ective. Edges where i has direct information have only one weight. Other edges can have two weights, corresp onding to the p ossibly con icting rep orts of the two agents involved.

In general, the true work graph is unknown to individual agents b ecause agents only have direct information ab out their own participation:

Deflnition 2. (Agent Information) Each agent i 2 V keeps a private history (w_i(i; j); w_i(i; j) and w_ii(j ; i)) of its direct interactions with other agents j 2 V , where w(j ; i) are the work performed for j and received from j respectively.

We assume that agents can share their information with each other, either via a centralized or a decentralized message exchange system. Based on its own exp eriences and the rep orts it has received from other agents, agent i can construct a sub jective work graph (see Figure 1). Let w_j(j ; k) 2 R_{≥0}(j ; k); w_k denote the edge weights as rep orted by agent j and agent k resp ectively. Definition 3. (Subjective Work Graph) A subjective work graph from agent i's perspective, G = (V_i ; E_i ; w_i) is a set of vertices V_i ⊆ V and directed edges E_i. Each edge (j ; k) 2 E_i for which i =2 {j ; k}, is labeled with one, or both, of weights w_ij(j ; k); w(j ; k) as known to i. For edges (i ; j) and (j ; i), the associated weight is w_ii(i ; j) = w_i(i ; j) and w_iik(j ; i) = w(j ; i) respectively. Note that the edge weights w_ij(j ; k) and w_k(j ; k) need not b e truthful rep orts ab out w (j ; k) and thus can p ossibly b e in con ict with each other, even if they have b een submitted at the same p oint in time. We now describ e in more detail how agents can exchange information with each other, to obtain the information necessary to construct the sub jective work graph.

Throughout the pap er, we analyze and compare two difierent mo des of information sharing b etween the agents: centralized and decentralized information exchange. For centralized information exchange, we assume the existence of a center (e.g., a centralized server on the Internet). After every interaction, each agent makes a rep ort to the center which stores all rep orts p ersistently. At any p oint in time, an agent can query the center to obtain the most up-to-date information ab out all rep orts available at the center to then construct its sub jective work graph based on his own information and the information obtained from the center. Note that every agent has difierent private information ab out his own interactions, and the other information is not necessarily correct, b ecause agents can make false rep orts to the center. Thus, the resulting sub jective work graphs for the agents can difier, even if all the rep orts are stored at the center. However, if agents i and j b oth rep ort truthfully to the center, then their resp ective sub jective work graphs G_i and G_j are the same. In many environments, a centralized information exchange system is simply not feasible, for example in wireless ad-ho c networks. In other environments, a centralized system might not b e desirable for many reasons: a center represents a single p

oint of failure, a center presents a bandwidth b ottleneck, and a center requires some a priori trust in one entity and it is unclear how that trust should b e established. This motivates the study of fully decentralized information exchange systems.

We now present BarterCast, a fully decentralized, lightweight information exchange system. In BarterCast, each agent also keeps a private history of its direct interactions with other agents, but

6

Figure 2: Accounting Mechanism and Allocation Policy: based on the subjective work graph G and the current choice set C_i, the accounting mechanism computes a score $S_{M_i}(G, C_i)$ for each agent in the choice set. Based on these scores, the allocation policy selects one agent for whom agent i will perform work.

in addition, an agent obtains information about the rest of the network by exchanging a selection of its private history with others using bilateral messages. We assume that peers can discover other peers with whom to exchange messages by using a Peer Sampling Service. When two agents agree to exchange messages, then agent i selects for its messages the records of the N agents with the highest amount of work performed for i as well as the N_{rh} agents most recently seen by i. Thus, in contrast to a centralized information exchange system, in BarterCast, each individual agent will always only have an incomplete view of the whole network, and a significant part of the information it holds will not be up-to-date. However, we will show in Section 5 that despite these drawbacks, the subjective work graphs each agent constructs based on the information available via BarterCast are su–cient to distinguish sharers from free-riders.
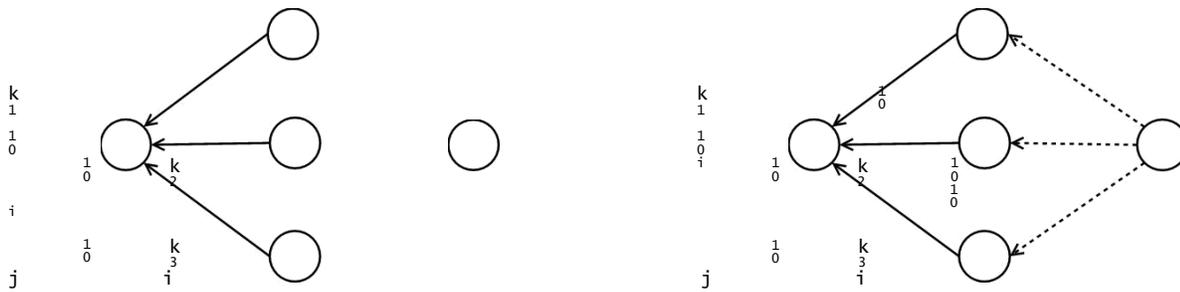
3 Accounting Mechanisms

3.1 Preliminaries

In the distributed work system, at every point in time, an agent can decide whether he is willing to perform work for others or not. An agent who makes himself available to perform work receives work requests by a set of agents (with which the agent may have rarely or never interacted with before). For example, in a P2P flle-sharing application, each agent that has any pieces of a particular flle will be contacted by a group of agents that are all interested in some of those pieces. At any moment in time, the contacted agent will have to choose for whom to perform work from this set of agents.

Deflnition 4. (Choice Set) We let $C_i \subseteq V \setminus \{i\}$ denote the choice set for agent i, i.e., the set of agents that are currently interested in receiving some work from i.

With agents $j, k \in C_i$, agent i must now decide whether he should do work for agent j or rather do work for another agent k. We assume that an agent has no a priori bias towards assisting one agent over another. We would like to know whether agent j is a "good" agent, i.e., whether he has done approximately as much work as he has received, or whether he is a "bad" agent that tries to free-ride on the system. The role of an accounting mechanism is to compute a score for each agent[1], proportional to the net work contributed by each agent $j \in C$ to the system to difierentiate between "good" and "bad" agents.

---

[1]Note that we purposefully chose to use the term "score" instead of "reputation value" even though this is in contrast to prior work by Meulpolder et al. (2009). Our goal is to clearly distinguish between accounting and reputation mechanisms and to emphasize that outputs of such mechanisms have very difierent meanings.

7

(a) (b) Figure 3: (a) A work graph based on true rep orts. (b) The sub jective work graph as viewed by i,
including a misrep ort attack by j to b o ost its score in BarterCast. Dotted edges indicate misrep orts.

Deﬁnition 5. (Accounting Mechanism) An accounting mechanism M takes as input a subjective work graph G_i, a choice set C_i, and determines the score $S_{Mj}(G_i; C_i)$, for any agent $j \in C_i$, as viewed by agent i.

Once the accounting mechanism has computed a score for each agent in the choice set, the agent uses an al location policy to decide who to allo cate work to (see Figure 2). Thus, the accounting mechanism together with the allo cation p olicy can b e seen as a matching algorithm, allo cating work-p erforming agents to work-seeking agents. In this pap er, we consider the following two allo cation p olicies:

Deﬁnition 6. (Ranking Policy) Given subjective work graph G_i, choice set C_i, and accounting mechanism M , agent i performs one unit of work for agent $j \in \arg \max_{k \in C_i} S_{Mk}(G_i; C_i)$, breaking ties at random.

Deﬁnition 7. (Banning Policy) Given subjective work graph G_i, choice set C_i, accounting mechanism M , and a banning threshold θ , agent i performs one unit of work for an agent chosen uniformly at random from $\{j \in C_j | S_{Mj}(G_i; C_i) \geq \theta \}$. Other allo cation p olicies are also conceivable, e.g., a prop ortional allo cation p olicy, but in this pap er,
we only compare the ranking and the banning p olicies. We will p erform a more detailed comparison of the two p olicies in Section 6, where we show the advantages and disadvantages of either p olicy in the P2P ﬁle-sharing domain.

3.2 Agent Population and Strategic Manipulations

We adopt the mo del and terminology of Meulp older et al. (2009), and assume a p opulation that consists of a mixture of cooperative agents (or sharers), who always contribute work, and lazy freeriders who intermittently shirk work. The role of an accounting mechanism is to make it unb eneﬁcial to b e a free-rider. We further mo del a subset of the free-riding agents as strategic agents, who also try to manipulate the accounting mechanism itself through misrep ort attacks, where an agent rep orts false information ab out its work p erformed or consumed. The non-strategic free-riders are called \lazy" b ecause they try to avoid p erforming work, but they are to o lazy to p erform any kind of manipulations. We also consider a second kind of attack on the accounting mechanism called sybil attacks where an agent inserts fake agents into the network to manipulate the mechanism. However, we defer a discussion of these attacks to Section 4.2.
8
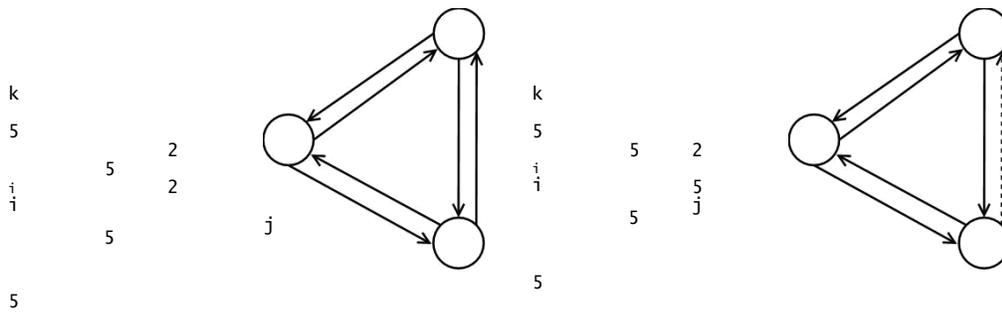
(a) (b) Figure 4: (a) A Work graph based on true reports. (b) The sub jective work graph as viewed by i, including a misrep ort attack by j to decrease k 's score and increase its own score.

i, any choice set C_iDefinition 8. (Misreport-proof) An accounting mechanism M is misreport-proof if, for any agent i 2 V , any subjective work graph G, any agent j 2 C, for every misreport manipulation by j , where $G_{0\ i}$is the subjective work graph induced by the misreports, the fol lowing holds:

† $S_{M\ j}(G_{0\ i};\ C_i) \bullet S_{M\ j}(G_{M\ k}(G_{0\ i};\ C_i)$, n f j

$S_{M\ ki}(G_i;\ C_i;\ C_i)$, and † S) 8k 2 C_i g.2

Given a misrep ort-pro of accounting mechanism, we assume that strategic agents will cho ose to follow the proto col and b e truthful. We assume that the accounting mechanism itself is flxed along with the allo cation rule of an agent. We mo del only strategic b ehavior with regard to rep orts ab out work p erformed or consumed. The motivating assumption is that agents are a priori indifierent as to whom they work for, with free-riders having preferences only in favor of receiving work and against p erforming work.

and choice set C_i

3.3 BarterCast Basic

Definition 9. (BarterCast Mechanism) Given subjective work graph G_i

B_i = $(V_i;\ E_i;\ w_{B\ i})$ with weights deflned as:

, construct a modifled graph G

$(j\ ;\ k\ )$ = $w_{i\ i}$
$B_{j\ i}(j\ ;\ k\ )$ = maxf $w(j\ ;\ k\ );\ w_{k\ i}(j\ ;\ k\ )g;$

; k g : w   8

(
j

;

k

)
j
i

———————————

2

f
j

(
j

;

k

)

8
(
j

;

k

perator are set
equal to 0. Let $MF^{B_i}_{G_{B_i}(\bot,j)_3}$

$G_{B_i}B_j(G_i; C_i) = MF$

2

3

9

(i,j)

denote the maximum ow from i to j in G. Deflne the BarterCast Score of agent j as $S(j ; i) ¡ MF$

In BarterCast, an agent takes its own information over rep orts from others. Given two rep orts, it takes the maximum of the two. Note that even if no agents misrep ort, two rep orts for the same edge will generally b e in con ict when a decentralized mechanism is b eing used due to the decentralized information exchange proto col. By taking the maximum of the two rep orts, an agent always uses the

Note that the flrst requirement is equivalent to value-strategyproofness as deflned for trust mechanisms, and b oth requirements together imply rank-strategyproofness [5].
This sp eciflcation of BarterCast difiers from Meulp older et al. [16] only in that they take the arctan of the difierence b etween the ows. However, b ecause arctan is a monotonic function this do es not change the ranking of the agents.

most up-to-date information it has. The max- ow algorithm b ounds the in uence of any rep ort that agent j can make by the edges b etween i and j , preventing an agent from grossly in ating the work it has p erformed for another agent. This limits the p ower of strategic manipulations and also protects against Byzantine attacks (i.e., arbitrary attacks not necessarily originating from rational agents). In some sense, using max- ow can b e seen as p erforming a form of bounded addition.

BarterCast can b e manipulated via misrep orts. In fact, it is a dominant strategy in BarterCast to always report 1 work performed, and 0 work consumed. We illustrate two attacks in Figures 3 and 4. We always show the sub jective work graph from i's p ersp ective and the manipulating agent is j . Figure 3 (a) shows a true work graph. Figure 3 (b) shows agent i's view of the work graph, now including a misrep ort by agent j . Agent j has simply rep orted that it has done work for k_1; k_2, and k_3, although it did not. BarterCast cannot catch this: b ecause there never was an interaction there are no rep orts from these other agents. Note that agent j increased its score from 0 to 30. Now consider Figure 4 (a) which shows a new true work graph. Figure 3 (b) shows a misrep ort manipulation by agent j where j rep orted that it has done 5 units of work for k even though it only did 2 units of work. Because BarterCast takes the maximum of two rep orts, agent i will b elieve j . As a result, agent k 's score has decreased from 0 to -3, and agent j 's score has increased from 0 to 3.

and choice set C_i

D_i
8(j ; k)j i 2 fj ; k g : $w_{D\,i}$(j ; k) = w_i:
$w_{D\,ii}$(j ; k) 8(j ; k)j j ; k 2 C

i; k 62 C_i: w (j ; k) = $w_{j\,i}$ i; C) = MF $_i^{GD}$
i; j 62 C_i j i(j ; k) = maxfw(j ; k); $w_{k\,i}$(j ; k)
i   :   g:
             D_j

             w
             D

$G_{D\,i}$(i; j).   i

          i   i
              10

set of agent i.

4 Theoretical Analysis   □
4.1 Information Loss of Drop-Edge
3.4 BarterCast + DropEdge
= (V, construct the modifled graph G
D_i(j ; k) = $w_{k\,i}$(j ; k) = 0 (1) 8(j ; k)j j 2 C
$w_{D\,i}$

Deflnition 10. (Drop-Edge Mechanism) Given subjective work graph G_i

   Missing reports in the max-operator are set to 0. Agent j 's score is S   (G $_{D\,i}$

     4   i

————
4

                                 (j ; i) ¡ MF

    Lines (1)-(3) implement a simple \edge-dropping" idea. Any rep orts received by agent i from agents in the choice set Care dropp ed in determining edge weights in mo difled graph G. An edge (j ; k) is

dropp ed completely if b oth j and k are inside C. We make the following observation: Prop osition 1. Drop-Edge is misreport-proof. Proof. No rep ort of agent j is used in i's decision making pro cess whenever agent j is in the choice

In this section we analyze the information loss of Drop-Edge due to the discarded edges. All statements are w.r.t. the centralized version of Drop-Edge only. The analysis is based on agent i's sub jective

Note that we do not need max- ow for misrep ort-pro ofness. However, we retain it b ecause it makes a more direct comparison with BarterCast p ossible and it protects against Byzantine attacks.

$= (V_i; E_i; w_i)$. $G_{D_i} = (V_i; E_i; w_{D_i})$ $O_i = (V_i; E_i; w_{O_i})$ $\sim w_i$ i

$i(k ; j)_i$
w_i

$i; E_i$
$c[work_i \ i$

$i(k ; G_i) =$
$D_i = (V_i \ P_{j 6 =}$
$ki; w_i)$
with
jV_i

work graph G_i $E (k ; G)] work) = 1 _i$) denotes the mo difled graph after the Drop-Edge mech
analogously for the omniscient mechanism (which adjus
oth Drop-Edge and the omniscient mechanism, we assu
manipulations. For the flrst theorem, we only consider th
Drop-Edge pro duces. We later add the use of max- ow
deflne the net work on edge $(k ; j)$ as $\sim w(k ; j) = w(j ; k$
Theorem 1. For al l subjective work graphs $G; E_j = n$, fo
uniformly at random with $jC j = m$ and $k 2 C$:

$i(k ; G_{O_i}$

$E_c[work_i(k ; G_c)] (4) = E[X_{j 6 = k} = X_{j 6 = k} = X_{j 6 =k} E_{\sim w_{D_i}}[\sim w_{D_i D_i}(k ; j)] (5)(k ; j)] (6)h (m _i 1)= _i 1 _i(n _i 1) \cent 0 +(m _i 1) (n _i 1)_i 1 _i\cent \cent work_i(m _i 1) (n _i 1)(k ; G_{O_i})\cent \cent \sim w_{O_i}$ For equation (7), consider edge $(k ; j)$. Because C is chosen uniformly at random with $k 2 C$, the probability that j is also inside any random C is. If k and j are inside C the edge gets dropp ed, otherwise $\sim w_{O_i m_j 1 n_i 1}(k ; j)$ is counted. Theorem 1 implies that if n is relatively large compared to m , then the exp ected net work computed by the Drop-Edge Mechanism is very close to the true net work.[5] The following corollary states this nice prop erty more formally:

Corollary 1. For al l subjective work graphs $G_i = (V_i; E_i; w_i)$ with jV_i
$[work^{O_{ii}}(k ; G_{D_i}) = 1.$

i

$D_{j; h}(G_i$

$D_{j; h}(G_i; C) = MF_{G_{D_i}; h}(j ; i) _i MF_{G_{D_i}; h}$

$G; h$

[5]

$(m _i 1)$ (7
$:$ $(n _i 1)$

Proof.
$(k ; j)_i$
$j = n$, for al l $k 2 V$, for choice sets C chosen
uniformly at random with $jC j = m$, it holds that:

m $E_c \ I$
i $!$
m 1
n

No 0 are reasonable. The exp ected ratio would then already b e ab ove 0.9995.

te 1 that Theorem 1 holds for any graph, including power-law graphs which we would exp ect in the file-sharing domain. There, a choice set size of 50 and a graph size larger than 100,000

] work (k; G

We now turn our attention to the approximation ratio of the scores computed by DropEdge when the max-flow algorithm is used. We consider max-flows restricted to a certain number of hops, and let $MF(i, j)$ denote the max-flow from node i to j in graph G with exactly n hops. We let $S(i, C)$ denote the score computed by DropEdge for n hops, i.e., $S(i, j)$. Analogously, $S_{G,h}(C)$ is the score computed by an omniscient mechanism for exactly n hops.

$= (V_i; E_i; w_i)$ with $jV_ij = n$, for al l $k \; 2 \; V_i$

$$\sum_{=1}^{h} \sum_{p=1}^{h} \frac{n_i 1 \; \textcent \; p \; \textcent \; \textcent \; S_{O} k \quad ; C}{;h(G \qquad):}$$

E_C

$+ \; n_i m_i \; 1h \; E_C \; [MF \; _{G_D i;h}(k \; ; i) \; \textcent \; MF \; _{G_D i;h}(i; k)] \; (14)$
$_{=2}X$

Theorem 2. For al l subjective work graphs G_i, for i's choice set $C = C$ chosen uniformly at random with $jC j = m$ a
$[S_D k(G_i; C)] = S_O k ; 0(G_i; \quad n_i m_i \; 1 \quad \prod_i n_i m_i p \qquad i$
$C) +$

Proof. $E_C \qquad [S_D k(G_i; C)] = \; [MF \; _i^{G_D} (k \; ; i) \; \textcent \; MF \; _i^{G_D} (i; k)] \; (8)$
E_C

$= \quad n_i m_i 1 X E_C [MF_{G_D i} \; ^{;h} \quad ^{G} \qquad _{G_D i;h}(i; k)] \; (9) \qquad\qquad (k \; ; i) \; \textcent \; MF$
$h = 0$

$= E_C[MF \; ^{G_D}_{i;0} \; (k \; ; i) \; \textcent \; MF \quad _{D i;0}(i; k)] \; (10)$

$+ \qquad E \quad n_i m_i \; 1h \quad ^{G} \qquad _{G_D i;h}(k \; ; i) \; \textcent \; MF \; _{G_D i;h}(i; k)] \; (11)$
$[MF \; _{=1}X$
$_{0}^{O k;}(G_i \qquad _{D i;1}(k \; ; i) \; \textcent \; MF \; _{G_D i; \; 1}(i; k)] \; (13)$
$c[MF$
$= S \; ; C) \; (12) + E$
$_{0}^{O k;}(G_i \quad ; C) + _i n_i \; m \; _{h=1} X_i \; m_i \; 1 \; n_i^{\textcent} \; \textcent \; S_O k ; 1(G_i; C) \; (15)$
$_{0}(G_i \quad 1 \; n_i \; 2n_i m_i \; 1h \qquad 2 \qquad _i$
$_{=3}X E_C$

$= S$

$$\prod_{p=1}$$
$+ _i \qquad \textcent \; \textcent \; n_i \; m_i \; 2 \quad \textcent \; \textcent \; S_O k ; \; ; C) \; (16)$
$_i \quad n_i \; 3 \qquad _2(G$
$+ \qquad [MF \qquad (k \; ; i) \; \textcent \; MF \; ^{G_D i;h} \qquad _{G_D i;h}(i; k)] \; (17)$
$= S^{O k;} \qquad n_i \; m_i 1 \quad ; C) \qquad \; ^h \; _i \; n_i \; m_i \; pn_i \; \textcent \; S_O k \quad ; C$
$_{0}(G_i \qquad + \qquad 1 _i \; p \; \textcent \qquad ;h(G \quad)$

In Equation 10 we isolated the exp ectation of the 0-hop max- ow terms, which are direct paths b etween i and k and thus the edges are not dropp ed and Equation 12 follows. In Equation 13 we isolated the exp ectation of the 1-hop max- ow terms which are paths of length 2 b etween i and k . Because C was chosen uniformly at random, the probability that the intermediate no de lies outside of C is $n_i m_i \; 1 \; n_i 2$ and Equation 15 follows. The flnal expression follows from analogous reasoning for all h -hop max- ows.

In practice, running the max- ow algorithm on the full work graph takes to o long. The following corollary tells us the accounting accuracy for a restricted max- ow algorithm:

$_{D k ;Hci}(G[SECorollary \; 2$. Let H denote the max number of hops used in computing max- ow. Then, for al l subjective work graphs $G_i = (V_i; E_i; w_i)$ with $jV_ij = n$, for al l $k \; 2 \; V_i$, for i's choice set $C = C$ chosen uniformly at random with $jC j = m$ and $k \; 2 \; C \; :; C)]$, $n_i m$
$_i H \; n_i \; H_¶ HS_O k ;H(G_i; C):i$

12

In the BarterCast implementation [16] and also in our exp eriments, the max- ow is restricted to at most 1 hop (i.e., paths of length at most 2). For that particular mechanism we get:
$$E_C[S_{D_{i;1}}(G; C)] \quad \text{¡} n \text{¡} m \text{¡} 1n \text{¡}$$
$$S_{O_{i;1}}(G_{ii}; C), \quad 2 \text{¢}:$$

Note that the theoretical results b ound the accuracy in exp ectation over choice sets and don't directly p ertain to accuracy with resp ect to selecting the right agent from a given choice set. Fortunately, the exp erimental results we present Sections 5 and 6 show that the information-loss of Drop-Edge is small and that the mechanism indeed p erforms very well in practice.

4.2 Sybil-pro ofness

Now we turn our attention to a class of attacks called sybil manipulations, where an agent intro duces sybil no des (fake agents) into the network to manipulate the accounting mechanism. For now, we will only consider sybil agents that do not p erform work themselves, but only try to consume work from other agents.

$$= (V_i; E_i; w_i \qquad\qquad s$$

$$s \quad 0 \quad M0$$
$$(e) = w_i(e) \text{ for } e 2 E_i \qquad\qquad (e) \text{ for } e 2 E_s \qquad\qquad = (V_i = G_0 \# \quad j^i \qquad\qquad [S;$$
$$13$$

$$M \qquad = G_{0i}$$

Deflnition 11. (Sybil Manipulation) Given a work graph G_i is a tuple $j = (S; E_s; w_s)$, a sybil manipulation by age agents created by agent j, $E = f(x; y): x; y 2 S [jg i$ the edges in E. Applying the sybil manipulation to G wand $w_0(e) = w_s$. We let Sdenote the default score t agent ab out which no information regarding work co Note that the analysis of a sybil manipulation do es Even though an attacking agent could do multiple th multiple false rep orts ab out these sybils, all of thes analysis only consider a graph G b efore the sybil m manipulation. Deflnition 12. (Sybil-Proofness) An ac subjective work

$$i = (V \quad j; E_i; w_i) \text{ and choice set } C_i, \text{ for any agent } j \text{ with any sybil manipulation } j = (V; E_s; \qquad w$$

$$i; C_i$$

raph G $C_{iiMs}(G_{0i}; C_i; C)$ $)$ such that $G\#$, the following three properties hold: (1) $S(G)$, $S_{Mj}(G)$ (2) $8k 2 V_n fj g: S_{Mgpi}(G;$ $)$ (3) $8s 2 S. S)$ • S_i) • $S_{M0Mk}(G_{0i}; C_i$

i.e., (1) j cannot increase its own score, (2) j cannot decrease another agent's score, and (3) j cannot create a sybil no de that has a higher score than the default score. It is easy to see that BarterCast and BarterCast+DropEdge are not sybil-pro of mechanisms. Consider Figure 5 where agent j p erforms a successful sybil manipulation.

We make the assumption that the scores an accounting mechanism computes only dep end on the amount of work p erformed and consumed by the agents in the network. More sp eciflcally, we assume that adding or removing agents with no amount of work consumed or p erformed do es not changes the scores of other agents:
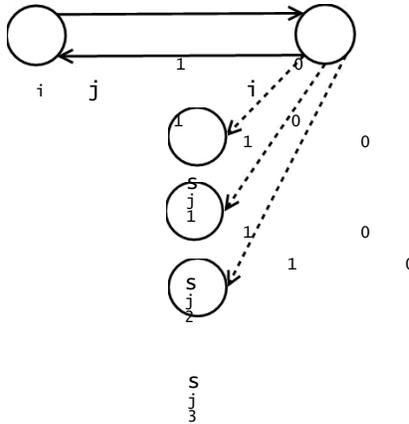
Figure 5: A sybil attack where agent j generates many sybils, then do es a little bit of work for i and then provides its sybils with p ositive reputation.

Assumption 1. (Independence of Disconnected Agents)) Given any subjective work graph $G = (V_i; E_i; w_i)$ and any choice set C_i, for any k 2 V_i for which there does not exist an edge in E_{ii} or for which al l edges in E_{ih} have zero weight, let $G_{0\,i} = (V)$ denote the graph where node k has been removed, i.e., V_0 i= V in f k g, E_0 i= E_{i0} i; E_0 i; w_0 in f(x; y) : x = k _ y = k g, and w_0 i(e) = w_i(e) for al l e 2 E_0 i, then we assume that the fol lowing holds:

$$0\,i8j\,2\quad : S_M\,(G_i; C_i) = S_{M\,j}\,(G\;\;;\;C_0$$
$$V \qquad j \qquad\qquad\qquad 0\,i \quad i)$$

We assume that, a priori, the accounting mechanism do es not put more or less trust into any agent in the network. We will only consider mechanisms which, for any renaming of the agents in the network, return the same scores:

Deflnition 13. (Symmetric Accounting Mechanisms) An accounting mechanism M is symmetric if, for any node i with any any subjective work graph $G_i = (\,V_i; E_i; w_i)$ and choice set C_0, any graph isomorphism f such that $G_i = f\,(G_i)$, $C_{0i} = f\,(C_i)$ and f (i) = i:i

$$8j\,2\,V_i\quad n\,fig : S_M\,(G_i; C_i) = S_{M\,f\,(j\,0}(G_i;\quad):$$
$$j \qquad\qquad) \qquad\qquad C_{0i}$$

We will exclude from our analysis any \trivial" accounting mechanisms, which excludes those mechanisms that assign the same score to every agent or that assign random scores to every agent. But we will also exclude mechanisms that ignore all information except for their own direct exp eriences. We b egin our formal analysis of sybil-pro of accounting mechanisms, assuming that even one rep ort from an agent can change the score of another agent. To make this more formal:

Deflnition 14. (Single-Report Responsiveness Property) Let dist(i; j) denote the distance between two nodes i and j in a graph, i.e, the length of the shortest path between those two nodes. An accounting mechanism M has the single-report responsiveness property if, for any agent i, there exists a subjective work graph $G_i = (\,V_i; E_i; w_i)$ and choice set C_i, with nodes j and k such that dist(i; j) = dist(j ; i) = 1 and dist(i; k) = dist(k ; i) = 1 (i.e., nodes i and j are neighbors in G and there is no path connecting nodes i and k), and there exists a graph G_0 i= (V_0 i; E_0 i; w_0 i) with V_0 ii= V_i, E_0 i= E[f(k ; j); (j ; k)g, and w_0 i(e) = w_i(e) for al l e 2 E_{ii} n f (k ; j); (j ; k)g , and there exists a constant c 2 $R_{>\,0}$ with $w_{0j\,i}(k\;;\,j\;) = c$, such that:

$$(G_{0;i}C_{0\,i}) > S_{M\,0}$$
$$S_{M\,k}\quad 14$$

What the single-rep ort resp onsiveness prop erty says is that there exists a situation, where a single p ositive rep ort by agent j ab out agent k increases the score that agent i assigns to agent k ab ove the default score.

Note that BarterCast-basic and BasterCast+DropEdge are b oth symmetric mechanisms that satisfy the single-rep ort resp onsiveness prop erty. We have already shown that b oth mechanisms are susceptible to sybil attacks. We will now show that this is generally unavoidable:

Theorem 3. No accounting mechanism M that satisfles Assumption 1, is symmetric and has the single-report responsiveness property is sybil-proof.

Proof. Let's assume accounting mechanism M satisfles the indep endence assumption (Assumption 1), is symmetric and has the single-rep ort resp onsiveness prop erty. Thus, there exists a graph G and no des i, j and k as describ ed in Definition 14. Now, let agent j create a sybil no de s_i and insert it into G_i such that $G_{0i} = (V_{0i}, E_i, w_i)$ with $V_{0i} = V_g$. Because of the indep endence assumption, the scores of all agents in the graph have remained the same. Note that there is no path connecting k and i as well as no path connecting s_i, i, s_i and i, and thus the two no des k and s_i lo ok the same from i's p ersp ective. Because M is symmetric, we could apply a graph isomorphism i to G that only switches the lab eling of s_i and k but nothing else. Thus, from the single-rep ort resp onsiveness prop erty, we can now conclude that there exists a rep ort that j can make ab out s_i with $w_{0j\,i}(s_i, j) = c$ leading to graph G_{00} such that $S_{M\,s_i}(G_{00\,i}, C_i) > S_{M\,d_j}$. This violates prop erty (3) of sybil-pro ofness in Definition (12), and thus M is not sybil-pro of.

One might argue that, even though an accounting mechanism M is not sybil-pro of, all p ossible sybil manipulations that it allows could b e non-beneflcial for the manipulating agent. For example, it is conceivable that, even though an agent j might b e able to create sybils with a p ositive score under mechanism M , the mechanism would in return decrease agent j 's score in such a way, that overall, the sybil manipulation would not b e b eneflcial for j . We will show next that, if we also assume that the accounting mechanism is misrep ort-pro of, then every mechanism with the ab ove prop erties has a b eneflcial sybil manipulation. Let's flrst formalize what we mean by a beneflcial sybil manipulation :

Deflnition 15. (Beneflcial Sybil Manipulation) Given any work graph $G_i = (V_i, E_i, w_i$

by agent $j \in V_i$ such that $G_{0i} = {}_j(G_i$

$_j\,{}_{M'j}(G_i, C_i)$ ${}_{M\,j}(G_{0\,i}, C_i$

(1) $9C_i$

$_i$ with j ; $k \in C_i$: $(S_{M\,j}$ ${}_{M\,g} \wedge (8C_i$ with $j \in C_i$: $S_{M\,j}(G_{0\,i}, C_i)$, $S_{M\,j}$ ${}_{0\,i}; C$

^{15} $_{i}; C_i$

: $S_{M\,s}(G_{0\,i}; C_i$

), a beneflcial sybil manipulation) is one where property (1), (2) or (3) holds:

$_i$ $_i$ $_i$ $_i$

(G

$_i$

)) (3) $9s \in S$: ($9C$ with $s \in C$) > S)) i.e., agent j can (1) increase
score and that of another agent
in such a way that the relative ranking of the two agents change^s
osen score while non-decreasing its own score. Note that man
ones that we worry ab out when designing reputation mechanisr
unique to the design of accounting mechanisms.

Theorem 4. For every accounting mechanism M that satisfles As
single-report responsiveness property and is misreport-proof, th
manipulation.

.t. $j \in C_i$ and S) (2) $9k \in V_n$ fj g and $9C$) < S

Proof. Consider the same set-up as in the proof for Theorem 3, with agents i, j, and k. Now, assume that agent k performs c units of work for i and agent j makes a truthful report to i about this interaction, leading to subjective work graph G. Because M is misreport-proof, we know that agent j is best-off making this report truthfully, i.e., for any possible misreport manipulation that would lead to subjective work graph G''_{0j} it holds that: $\exists C_i$ with $j \in C_i$: $S_{Mi}(G_0^i, C_i) \leq S_{Mi}(G)$. Now, analogously to the argument in the previous proof, because M satisfies the independence assumption and is symmetric, we know that agent i also does not decrease its own score (when it is inside the choice set) when j reports to i that sybil s_i has performed c units of work for j, i.e., $w_{0j}(s_{j00i}, C_i, j) = c$. But we have already shown in the previous proof, that due to the single-report responsiveness property, this report leads to a score of $S_{Msi}(G_0^{0i}, C_i) > S_M 0$, providing sybil agent s_i with a positive score, while non-decreasing the score for agent j (whenever j is inside the choice set). This constitutes beneficial sybil manipulation number (3), which completes the proof.

The only property that we can reasonably relax for the design of useful accounting mechanisms is the single-report responsiveness property. We can conceive of mechanisms that require two, or more generally, k positive reports about an agent, before the mechanism assigns a score strictly larger than $S_M 0$ to that agent. If this is the case, then the simple sybil manipulations we have used for the proofs of the above theorems don't work anymore. For a successful sybil manipulation against such mechanisms, the sybil agents themselves would have to become active, and perform some work in the network, such that at least another agent (or more generally, $k-1$ other agents) makes a positive report about the sybil agent. While this would lead to some notion of k-sybilproofness, these mechanisms would also have to ignore a much larger part of the information, because in a distributed system, the probability of receiving two, or more generally, k reports about the same agent decreases exponentially. Thus, in practice, there would be a trade-off between robustness against sybil manipulations on the one side, and informativeness on the other side. Analyzing this trade-off in more detail (analytically and experimentally) is subject to future work.

The importance of robustness against sybil attacks also depends on the domain, i.e., in particular how difficult it is for the agents to perform a sybil manipulation. In P2P networks, for example, misreport attacks are much easier to execute than sybil attacks and also more beneficial. Sybil attacks might require adopting multiple IP addresses and coordinating work between them. According to Pouwelse et al., they have not yet seen sybil attacks in practice, while misreport attacks have a long history (c.f. Kazaa Lite, eDonkey). Thus, for practitioners of P2P file-sharing networks, protecting against misreport attacks is generally of higher importance than protecting against sybil attacks.

4.3 The Role of the Max-Flow Algorithm

We will now briefly discuss the motivation for using the max-flow algorithm inside BarterCast and BarterCast+DropEdge to compute agents' scores. Ideally, we would like to do accounting via taking the total sum of work performed and subtracting the total sum of work consumed for each agent. By using the max-flow algorithm, we essentially do a form of \bounded addition" which obviously distorts the true net work measure. The BarterCast+DropEdge mechanism would also be misreport-proof without the use of the max-flow algorithm, however, there are multiple reasons for using max-flow. First, it is useful to protect against Byzantine agents, i.e., agents that are not necessarily acting rationally but who simply try to harm the network or specific agents in the network. If such a Byzantine agent reports that agent i has consumed 1,000,000 units of work from him, and if other agents believe this report, then agent i will be unable to receive any work from those agents in the future and there is nothing i could do about it. Using max-flow bounds the influence of any agent by

16

the amount of work it has p erformed/consumed in the network, and thus makes Byzantine attacks much more di–cult and costly for the attacking agent, thereby efiectively preventing them in practice.

A second reason for using max- ow is to protect against sybil and collusion attacks. As we have discussed in the last section, a fully sibyl-pro of accounting mechanism would have to discard to o much information, and thus in practice, non-sybil-pro of accounting mechanisms might b e used. However, by using the max- ow algorithm, we b ound the in uence of any agent, thereby limiting the p ower of sybil attacks, making them more costly and thus less attractive for the attacking agent. Note that instead of using max- ow, we could also use other graph-based algorithms. The algorithm only needs to have two prop erties: flrst, it needs to have the \b ounding prop erty" to limit the in uence of any agent prop ortionally to how much that agent has contributed to the system so far. Second, the algorithm must have the \transitive-trust" prop erty, i.e., when agent i has p erformed some work for j and j has p erformed some work for k , then i should also trust agent k to some degree. We refer the interested reader to [23] where we review the most imp ortant transitive-trust mechanisms (shortest-path, EigenTrust, PageRank, Max-Flow, Hitting-Time) and intro duce a new class of hybrid transitive-trust mechanisms, which could also b e used for the design of accounting mechanisms.

5 Exp erimental Analysis: Discrete Simulations

In this section, we evaluate the mechanisms empirically via simulations to b etter understand the trade-ofis that are made in the Drop-Edge and BarterCast mechanisms. We b egin our analysis with a discrete, round-based simulation where agents p erform discrete units of work in each round. In Section 6 we move on to a more realistic simulation, using the accounting mechanisms as an overlay proto col for a BitTorrent client.

5.1 Exp eriment Set-up

Here, we simulate a P2P flle-sharing environment with 100 agents and discrete time steps. Downloading a flle corresp onds to consuming work and uploading a flle corresp onds to p erforming work. In every time step, every agent decides whether to p erform one unit of work or not. Agents are divided into a fraction 1 ¡ fl of co op erative and a fraction fl of free-riding agents. Co op erative agents always p erform one unit of work, while free-riders only p erform work in every other round. Furthermore, we also mo del strategic free-riding agents who seek to manipulate the accounting mechanism. We let   • fl denote the total fraction of all agents that are strategic free-riders. With BarterCast, the strategic agents p erform the optimal misrep ort manipulation, i.e., always rep orting they have consumed 0 units of work and contributed 1 units of work.In each round that agent i p erforms work, it gets a random choice set of 5 agents. With probability 0 :1, i p erforms 1 unit of work for a random agent in the choice set, and with probability 0.9 it uses the accounting mechanism and allo cation rule to determine who receives work. This simulates the \optimistic unchoking" found to b e useful in distributed work systems such as BitTorrent. Each round, every agent contacts one other agent at random to exchange messages ab out their direct exp eriences in the network. All agents send a rep ort ab out the last 5 agents they have interacted with and the 5 agents that have uploaded the most to them. Strategic agents are untruthful when sending these rep orts.

For b oth Bartercast and Drop-Edge we run a centralized and decentralized version for the exp eriment. In the centralized version, all rep orts (which can still b e untruthful) are made directly to a central entity, immediately after each interaction, and are then available to every agent. Considering the centralized version of each mechanism helps isolate the efiect of the message passing algorithm. We
17

run each simulation for 100 time steps and record the work contributions and consumptions (averaged over 10 trials).
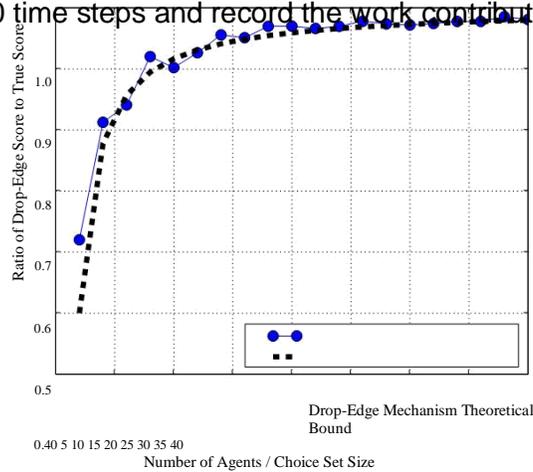

Figure 6: The approximation ratio of the scores.

5.2 Information Loss of Drop-Edge

We flrst verify our theoretical results on information loss from Section 4.1. Fixing a choice set size m = 5, free-rider agent fraction fl = 0:5, and strategic agent fraction = 0:2, we simulate networks of size n = 10 ; 20 ; :::200. After 100 time steps, for every agent we randomly cho ose a choice set and measure for every agent in the choice set the ratio of the Drop-Edge score and the score under the omniscient mechanism. Averaging over all agents and choice sets, we flnd that our empirical results closely match the theoretical results from Corollary 2 (see Figure 6).

5.3 E–ciency Results

Now that we have established exp erimentally that at least on average, Drop-Edge provides a go o d approximation to the agents' scores, we turn our attention to directly measuring the mechanisms' p erformance. First, we measure their p erformance without strategic agents, to isolate their efiectiveness as algorithms in aggregating information and promoting go o d decisions. Consider the graphs in Figure 7 (a) with zero strategic agents, i.e. where = 0 . We exp ect Drop-Edge to b e slightly less e–cient b ecause we are dropping information that BarterCast is using, and no strategic agents are present that could harm BarterCast. We see that the e–ciency is indeed higher under b oth versions of Bartercast, but only minimally so (less than 5% difierence).

   The more interesting analysis concerns the overall e–ciency with strategic agents present. The e–ciency of a particular agent typ e is deflned to b e the average amount of bandwidth received by that typ e of agent p er time step. It is our goal to maximize the e–ciency of the cooperative agents and to minimize the e–ciency for free-riding agents, and for strategic free-riders in particular. Ultimately, the goal is to cause agents to change from free-riding to co op erating.

   We compare Figures 7 (a),(b) and (c), to analyze the relative e–ciency of all agent typ es under the Bartercast and Drop-Edges mechanisms. Note that the total e–ciency is the same for b oth mechanisms b ecause the amount of work p erformed by individual agent typ es is flxed. In Figure 7(b), we clearly see that strategic agents are able to sharply increase their p erformance compared to the other agents (see Figures 7(a) and (c)) by misrep orting under the BarterCast mechanism. This efiect is particularly high when only a few strategic agents are in the system. With 10% strategic agents, the

18
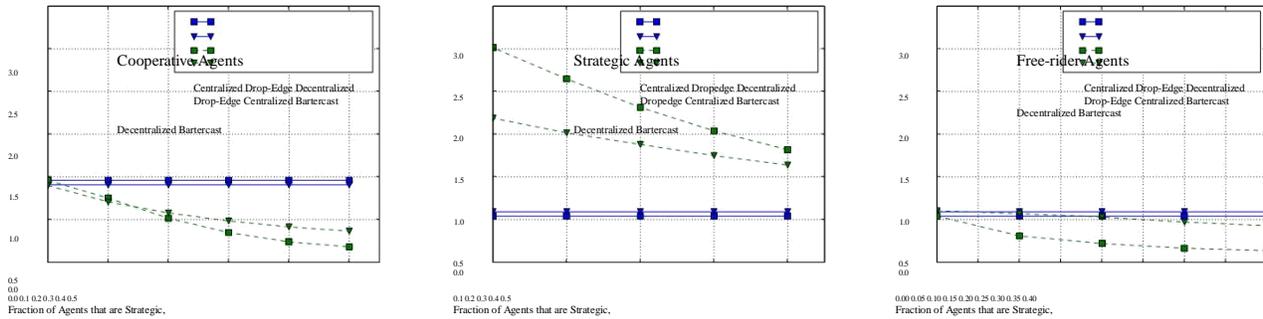
(a) (b) (c) Figure 7: Bandwidth of difïerent typ es of agents, using Bartercast and Drop-Edge. The fraction of
free-riding agents is fl = 0:5.

p erformance of a strategic agent is 3 times as high as that of the other agents under the decentralized BarterCast mechanism, and more than 5 times as high under the centralized BarterCast mechanism. With BarterCast, agents have a very large incentive to act strategically. The Drop-Edge mechanism in contrast leads to the same constant e–ciency for each individual agent typ e (b ecause there is no incentive to manipulate), and in particular the e–ciency of co op erative agents is almost twice as high as that of free-riding agents.

In practice, strategic misrep orts may also o ccur under Drop-Edge even though such b ehavior is not rational for an individual agent. We have tested Drop-Edge in settings with strategic agents (not plotted) and although the e–ciency of the co op erative agents decreases slightly as the prop ortion of strategic agents increases, Drop-Edge continues to clearly outp erform Bartercast. Interestingly, if we move from a 1-hop max ow to a standard max ow algorithm, the e–ciency of co op erative agents under Drop-Edge actually increases as the prop ortion of strategic agents increases. A more detailed analysis of this efiect is sub ject to future investigations.

We also ran a longer exp eriment with fl = 0:5; = 0 :2 for 500 time steps, measuring how e–ciency changes over time. In Figure 8 (a), we see that the b eneflt that strategic agents gain from misrep orting in BarterCast gets even larger over time. Compare this against Figure 8 (b), which presents results for DropEdge. Strategic agents cannot manipulate their scores, and receive decreasing amounts of work as the simulation pro ceeds. At the end of the run, co op erative agents indeed receive twice as much work p er round as the other agents (note they also p erform exactly twice as much work).

## 6 Exp erimental Analysis: A BitTorrent Overlay Proto col

In this section we discuss our application of BarterCast and BarterCast+DropEdge in BitTorrent, a p opular P2P flle-sharing proto col. BitTorrent currently only implements a tit-for-tat p olicy which gives downloaders in a single swarm an incentive to upload to each other. However, there is no incentive to continue sharing the flle after the download has flnished. Ironically, it is even disadvantageous to share, since the consumed upload bandwidth cannot b e used to do tit-for-tat in other downloads, which makes these downloads slower. Using accounting mechanisms on top of the existing BitTorrent proto col, we want to remove this incentive problem to increase the overall e–ciency of the system.

The normal BitTorrent proto col maintains a limited numb er of simultaneous upload slots (usually 4-7 dep ending on the implementation). Peers that do not yet have the complete flle (leechers), assign their slots to those p eers that currently provide the highest upload rate in return, determined p erio dically. Peers that have the complete flle (seeders) assign their upload slots to those p eers that

19

Average Bandwidth per Timestep

Average Bandwidth per Timestep

Average Bandwidth per Timestep
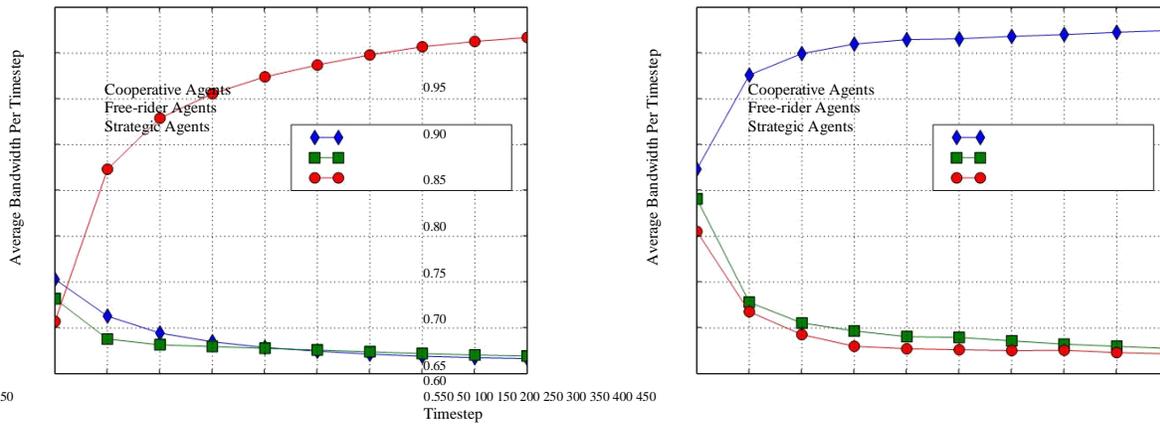
(a) BarterCast (b) Drop-Edge Figure 8: Bandwidth in BarterCast and
BarterCast+Drop-Edge over Time.

have the highest download rate. Peers that get a slot are called unchoked, while the other p eers are choked. Furthermore, there is one extra slot for optimistic unchoking. This slot is assigned via a 30 seconds round-robin shift over all the interested p eers regardless of their upload rate. The proto col therefore creates a tit-for-tat data exchange based on the short-term b ehaviour of a p eer (i.e., the bandwidth it provides in return). Due to optimistic unchoking, new p eers have a chance to obtain their flrst pieces of data and b o otstrap the pro cess.

6.1 Accounting Mechanisms in BitTorrent

When an accounting mechanism is available in a P2P flle-sharing system such as BitTorrent, this raises the question which allo cation p olicy to use. A natural candidate would b e the ranking p olicy which always gives preference to agents with a higher score. As we have shown in Section 5, in the discrete simulations the ranking p olicy works very well, and successfully separates sharers from free-riders. However, the situation is more complicated when we are targeting a system like BitTorrent that is already deployed in practice. First of all, a new BitTorrent client should b e backwards-compatible with old BitTorrent clients that are not using the accounting mechanism. Secondly, a user who installs the new BitTorrent client with the accounting mechanism should have a p erformance at least as go o d as with the old BitTorrent client. But this puts some restrictions on what kind of allo cation p olicies we can usefully employ.

   Imagine agent i using BarterCast in a BitTorrent network with primarily \standard BitTorrent clients", i.e, those that do not use the accounting mechanism. A standard BitTorrent client allo cates the optimistic unchoking slot to a random agent. Thus, if agent i uses the accounting mechanism to decide which agent to unchoke optimistically, this do es not afiect its p erformance. However, the remaining upload slots are normally allo cated to those agents providing the highest download sp eed in return. Myopically, this optimizes the download p erformance for the uploading agent. Now, if agent i would allo cate al l upload slots based on the accounting mechanism, agent i's p erformance could degrade, b ecause p ossibly the agents with the highest scores have uploaded a lot in the past, but they do not have any pieces to recipro cate in the current swarm. Thus, agent i would b e worse ofi using BarterCast than using the standard BitTorrent client, and this must b e avoided.

   Consequentially, the ranking p olicy that we employ in our exp eriments with BitTorrent only uses the accounting mechanism to decide who to allo cate the optimistic unchoking slot to. The second

20

p olicy we employ is the banning p olicy where an agent will never upload to an agent with a score b elow a certain threshold. The idea here is that the threshold is set low enough such that sharers will never reach a score lower than this threshold. Thus, banning an agent completely that has a score b elow the threshold only excludes the free-riders from the network. Note that b oth p olicies are easy to implement on top of any P2P flle-sharing proto col and backwards-compatible with existing mechanisms:

† ranking p olicy: Peers assign optimistic unchoke slots to the interested p eers in order of their scores. Therefore, a p eer can not get an upload slot while p eers with a higher scores are also interested and not yet served.

† banning p olicy: Peers do not assign any upload slots to p eers that have a score which is b elow a certain threshold – .

6.2 Simulation setup

6We have built a simulator which incorp orates all relevant asp ects of BarterCast and BitTorrent. We simulate an epidemic Peer Sampling Service combined with the BarterCast proto col and the BitTorrent proto col.Our BitTorrent simulator follows the proto col at the piece-level, including unchoking, optimistic unchoking, and rarest-flrst piece picking. We have combined all pro cesses in simulation environment that can generate workloads based on either probabilistic request arrivals or data of traces.

In our exp eriments we simulate 100 p eers active in 10 swarms (i.e., corresp onding to 10 difierent flles) during a simulated time of one week. Note that a p eer can b e active in multiple swarms simultaneously. A p eer can b e either a sharer, a free-rider, or a strategic peer. free-riders immediately leave the swarm after flnishing a download, while sharers share every downloaded flle for 10 hours. Strategic p eers b ehave as free-riders, and in addition spread only forged BarterCast messages in which they rep ort a maximum upload to others and zero download. All p eers in our simulations have a 3 MBps downlink and a 512 KBps uplink, corresp onding to common ADSL users. As these users have very limited uploading capacity, they are likely to economize on sharing, and are therefore the most imp ortant target of sharing-ratio enforcement in current flle-sharing systems. Finally, in the BarterCast messages we have used $N_h = N_r = 10$. Using our simulator, we p erformed two difierent typ es of exp eriments. First, we simulated the arrival pro cess of agents in the network according to a p oisson distribution. Second, we did tracebased simulations where the b ehavior of the agents was mo delled closely after traces from a real BitTorrent tracker.
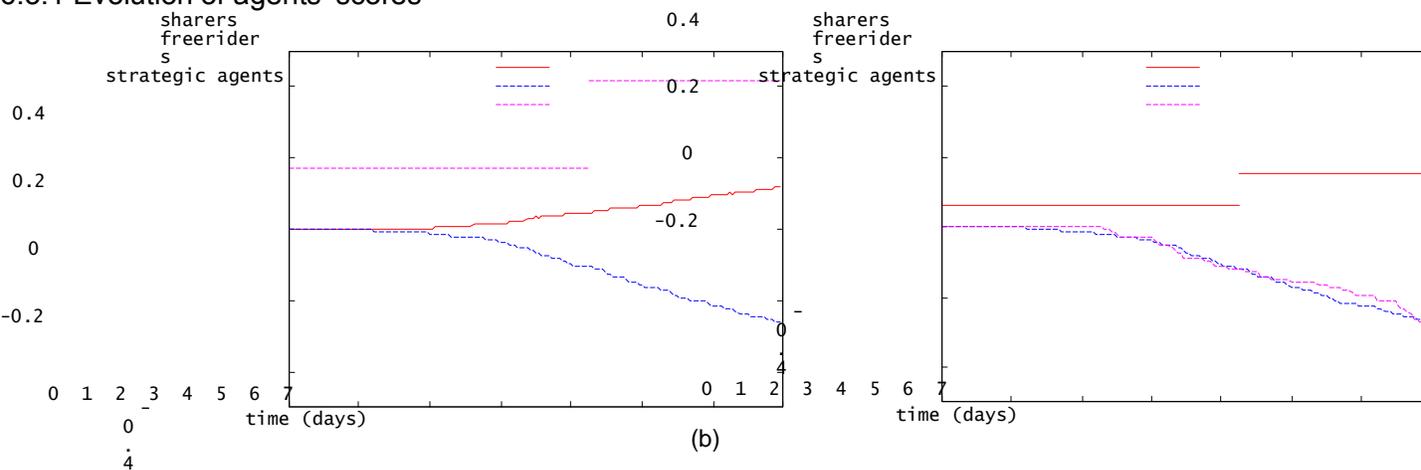
6.3 Poisson-based simulations

We implemented a random generator that for each p eer generates Poisson arrivals of flle request with an average of one request p er day p er p eer. Requests are homogeneously distributed over the 10 flles. In these simulations, 50% of the p eers are sharers, 40% of the p eers are free-riders, and 10% of the p eers are strategic p eers. In our exp eriments, we assess the evolution of the average scores of each group of agents, and compare the download p erformance for difierent p olicies. Furthermore,

---

6The actual implementation of the Peer Sampling Service is transparent to BarterCast, and it is su–cient here to note that distributed PSS proto cols are feasible and do exist. We use the decentralized BuddyCast epidemic proto col that is already implemented and released as part of the Tribler flle-sharing client [19] since Version 4.2

we evaluate the in uence of the threshold under the banning p olicy, and evaluate a simple mo del for b ehavioral change.

6.3.1 Evolution of agents' scores

sharers
freeriders
strategic agents

0.4

0.2

0

-0.2

0 1 2 3 4 5 6 7
time (days)

-0.4

(a)

0.4

sharers
freeriders
strategic agents

0.2

0

-0.2

-0.4

0 1 2 3 4 5 6 7
time (days)

(b)

$$\sum_{j} S_i(G_j; C_j) \quad (18)$$

In Figure 9 we have plotted the average scores of the sharers, the free-riders, and the strategic p eers over time during the simulation. Figure 9a shows the results for the scores based on max ow, without the use of Drop-Edge. The scores of the free-riders clearly decrease over time, while the scores of the sharers increase slightly. However, the strategic p eers b eneflt a lot from manipulating the accounting mechanism; their scores are signiflcantly higher than the scores of the sharers. This efiect vanishes completely when the Drop-Edge strategy is applied, which is displayed in Figure 9b. Here, the strategic agents have the same average scores as the free-riders. Obviously, the Drop-Edge strategy is very successful in reducing the p ossibilities for individual p eers to manipulate the mechanism.

6.3.2 Policy evaluation

We have compared the efiect of the banning and the ranking p olicy on the average download p er-formance of the p eers. All results are normalized with resp ect to the results of simulations that do not implement any p olicy at all. In order to distinguish the efiects of max ow, decentralization, and Drop-Edge, we have used the following set of accounting mechanisms:

† Central Max ow The actual up- and download statistics of p eers are stored in a central database, instead of in lo cal databases. The score $R_i(j)$ of p eer j in the eyes of p eer i is

22

Figure 9: (a) The average system scores without Drop-Edge; (b) The average system scores with Drop-Edge.

As a flrst step, we evaluated how the scores of the difierent agents evolved over time, i.e., whether the accounting mechanism can successfully track agents' net contribution to the system. Because each agent computes a difierent score for each other agent in the network, we compute the system score S of agent i as the average of the score that each of the other $N − 1$ agents assign to i:
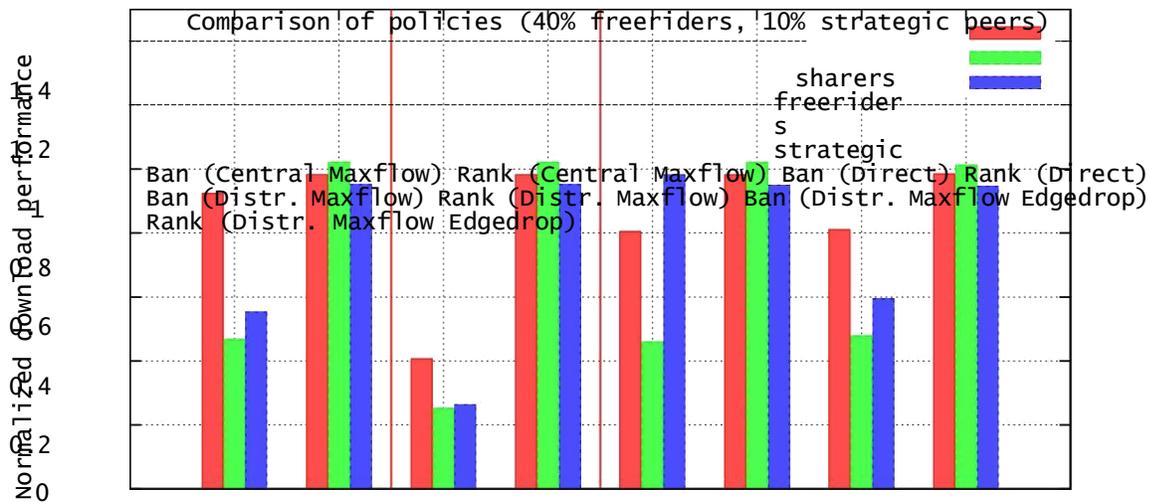
$$S_i = \frac{1}{N-1} \sum$$

Figure 10: Comparison of the average download p erformance of sharers, free-riders, and strategic p eers for b oth the banning and ranking p olicy with difierent accounting mechanisms.

computed using the max ow algorithm and the score metric, based on the central information. Note that while the statistics are centrally stored and ob jective, the computed score of p eer j dep ends on the p osition of p eer i in the network and is therefore still sub jective.

[7]† Direct Instead of computing the max ow over all paths from one p eer to another p eer, a p eer i computes the score of p eer j based on only the up- and download volume of direct transfers b etween i and j. † Distributed Max ow The max ow algorithm applied by each p eer based on a lo cal databaseof up- and download statistics obtained by BarterCast messages. † Distributed Max ow Drop-Edge The same as Distributed Max ow, with in addition theDrop-Edge strategy.

We have applied b oth the banning and the ranking p olicy to each of the ab ove accounting mechanisms. For the banning p olicy, the default banning threshold – is set to ¡0:5. In Figure 10, the normalized average download p erformance of the sharers, the free-riders, and the strategic p eers is displayed for each of the resulting combinations of accounting mechanism and allo cation p olicy. An
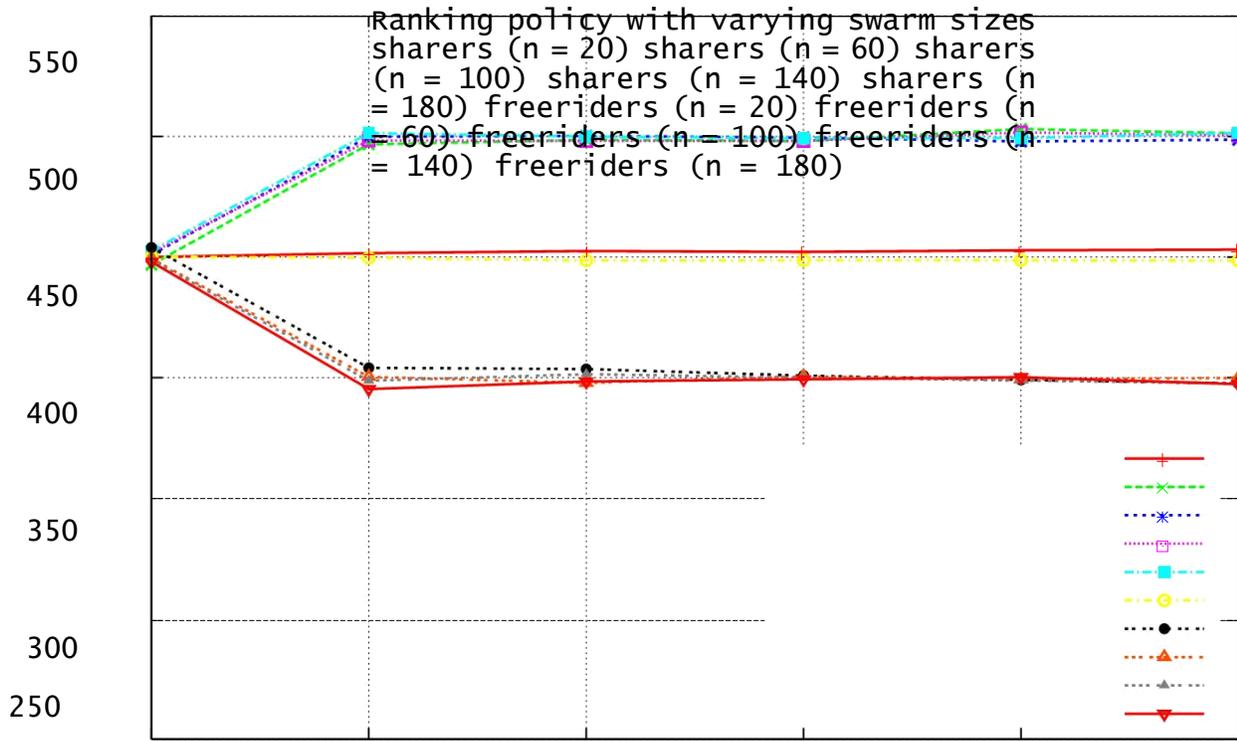
[7]Technically, this is equivalent to the max ow algorithm with a maximum path length of 1.

23

immediate and very interesting observation is that regardless of how the score is computed, the ranking p olicy has no signiflcant efect at all. An investigation of the results showed that this is caused by the size of the swarms. As we simulate relatively small swarms, p eers have not always requests of other interested p eers to flll all of their upload slots. Hence, free-riders can often flnd enough free slots to still have a normal p erformance. We argue that a p olicy based on ranking would only b ecome efiective if swarms are relatively large and p eers know a signiflcant fraction of the other p eers in the network. Only then would each p eer receive so many requests from other interested p eers that the free-riders are outranked by the sharers.

To verify this hyp othesis, we ran more exp eriments with varying swarm sizes. In Figure 11 we display the results from that exp eriment. On the x-axis we vary the accuracy of the accounting mechanism, where an accuracy of 0:4 means that, on average, the agents have accurate scores for 40% of the agents in the network, and none for the rest. First, it is easy to see that once the accuracy increases from 0% to 20%, the resulting p erformance do esn't change anymore. This is a p ositive result, b ecause it implies that even in a large system where agents will always only have a partial view of the network, the ranking p olicy works well as long as each agent has some minimal amount of information ab out other agents. Now let's consider the efect of changing the swarm sizes. As we can see, indeed, with a very small swarm size of only 20 agents, the ranking p olicy is not efiective in separating sharers from free-riders. However, as we increase the swarm size from 20 to 60 agents, this changes, as now the sharers have an average p erformance of 500Kbps and the free-riders have an average p erformance of 400Kbps. Thus, this verifles our previous hyp othesis that the ranking p olicy is only efiective for swarm of some minimal size.

However, Figure 11 also shows that the p erformance difierence b etween sharers and free-riders do es not increase further as we increase the swarm sizes from 60 to 180. It turns out, the explanation for this efiect lies in the details of the BitTorrent proto col. In our simulations, each agent has flve upload slots. We use the accounting mechanism to allo cate the optimistic unchoking slot. The other four slots are allo cated based on the standard tit-for-tat p olicy, and scores from past contributions in the system don't matter. Thus, sharers only b eneflt from the allo cation of the optimistic unchoking slot, which represents one out of flve slots. Thus, it is not surprising that the p erformance of the sharers is exactly 5 to 4 (i.e., 500Kbps to 400Kbps), the same ratio as the upload slots that are allo cated to them on average. This reveals an inherent limitation of using the ranking p olicy in BitTorrent. Because we want to maintain backwards compatility, and make sure that users of our mechanism are at least as well of as users of the standard BitTorrent mechanism, the maximal efiect that employing the ranking p olicy could have is limited: we can at most reduce the p erformance of the free-riders by 20% compared to the sharers. In most environments, this won't b e enough to incentivize the free-riders to change their b ehavior and b ecome sharers. Thus, we will investigate the p ower of the banning p olicy in more detail.

The results for the banning p olicy in Figure 10 show several very interesting prop erties of the various accounting mechanisms. First of all, the banning p olicy applied to the centralized max ow implementation causes the p erformance of b oth free-riders and strategic p eers to b e roughly cut in half. Obviously, as there are no messages involved, the strategic p eers are not distinguishable from the free-riders, so the difierence b etween those two groups is caused by random efiects. When the banning p olicy is applied to only direct information, not only the p erformance of the sharers and free-riders is very low, but also that of the sharers. This can b e exp ected; while a sharer has uploaded a lot in total, in its individual relationship with another p eer it might have downloaded much more than uploaded. Hence, a signiflcant fraction of the p eers might still ban a particular sharer. Even though the free-riders and strategic p eers have the lowest p erformance, the system as a whole sufiers signiflcantly when only direct information is taken into account. When the max ow algorithm is
24

Ranking policy with varying swarm sizes
sharers (n = 20) sharers (n = 60) sharers (n = 100) sharers (n = 140) sharers (n = 180) freeriders (n = 20) freeriders (n = 60) freeriders (n = 100) freeriders (n = 140) freeriders (n = 180)

Figure 11: Performance of

the Ranking Policy with different swarm sizes.

applied based on distributed information,

the results are much more attractive. The penalty for free-riders is comparable to that of the central approach. The sharers also experience a somewhat lower performance, but this trade-off might pay off in the end when free-riders would become sharers because they experience a low performance when they do not (see

Section 6.3.4). Finally, the efiect of the Drop-Edge strategy is once again very clear: the strategic p eers p erform b etter than the sharers if Drop-Edge is not used, and they have a p erformance comparable to the free-riders if is used. This is consistent with the evolution of their average score shown in Figure 16.

6.3.3 Influence of the threshold

We have investigated the influence of the banning threshold on the effectiveness of the banning policy. In Figure 12, the normalized download performance for sharers, free-riders, and strategic peers is plotted for various thresholds using the Central Maxflow, Distributed Maxflow, and Distribu

ted Max ow Drop-Edge implementations. In Figure 13, we have displayed the same results for the freeriders and strategic peers, relative to the download performance of the sharers. The latter figure shows clearly that the more strict the threshold (i.e., closer to 0), the more severe the relative penalty of the free-riders. Strategic p

eers have no beneflt when the Drop-Edge strategy is used. While from this perspective, the strictest threshold seems to be the best option, Figure 12 makes clear that a stricter threshold also gives a greater loss of performance for the sharers. There is no clear optimum. 2

Average download performance (Kbps)

1 0.8 0.6 0.4 0.2 0sharers freeriders strategic40% freeriders, 10% strategic peers, central maxflow-0.9 -0.8 -0.7 -0.6 -0.5 -0.4 -0.3 -0.2 -0.1 normalized download performance
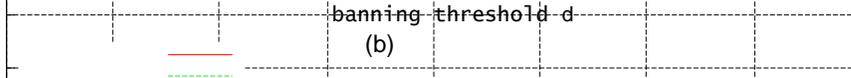
banning threshold d
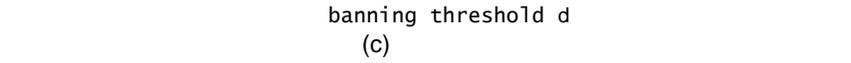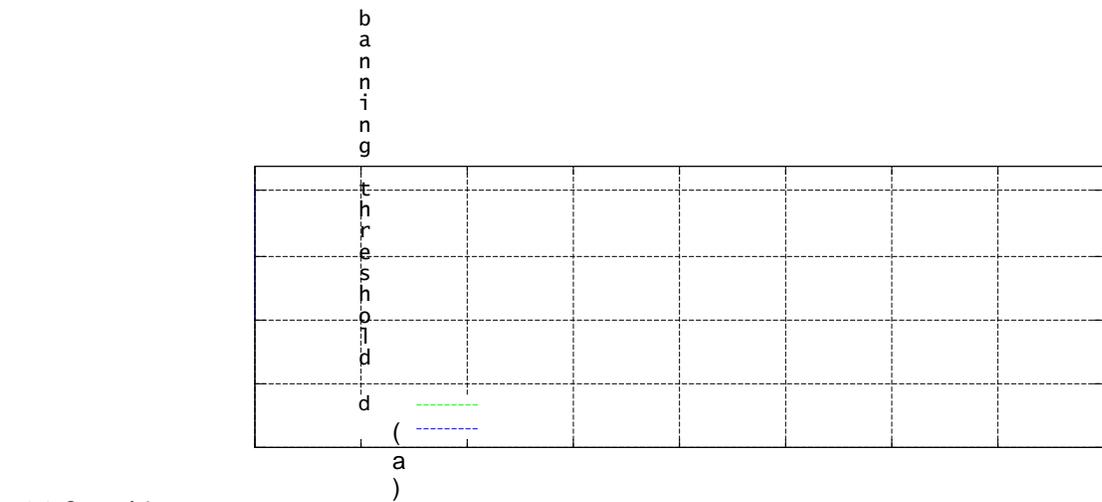
(a)

0 0.2 0.4 0.6 0.8 140% freeriders, 10% strategic peers, distributed maxflowsharers freeriders strategic-0.9 -0.8 -0.7 -0.6 -0.5 -0.4 -0.3 -0.2 -0.1 normalized download performance

banning threshold d

(b)

0 0.2 0.4 0.6 0.8 140% freeriders, 10% strategic peers, distributed maxflow edgedropsharers freeriders strategic-0.9 -0.8 -0.7 -0.6 -0.5 -0.4 -0.3 -0.2 -0.1 normalized download performance
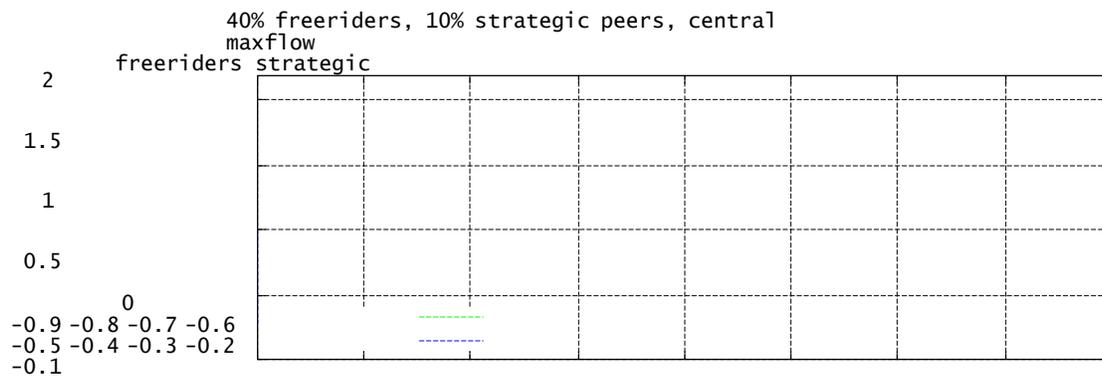
banning threshold d

(c)

Figure 12: The normalized download p erformance of sharers, free-riders, and strategic p eers using the banning p olicy for various thresholds – under (a) Central Max ow, (b) Distributed Max ow, and (c) Distributed Max ow Drop-Edge.

26

40% freeriders, 10% strategic peers, central maxflow

freeriders ————
strategic ————

2
1.5
1
0.5
0
-0.9 -0.8 -0.7 -0.6
-0.5 -0.4 -0.3 -0.2
-0.1

banning threshold

freeriders ————
strategic ————

(a)

40% freeriders, 10% strategic peers, distributed maxflow

freeriders ————
strategic ————

2
1.5
1
0.5
0
-0.9 -0.8 -0.7 -0.6
-0.5 -0.4 -0.3 -0.2
-0.1

banning threshold

40% freeriders, 10% strategic peers, distributed maxflow edgedrop

d (b)

freeriders
strategic

2

1.5

1

0.5
0

-0.9
-0.8
-0.7
-0.6
-0.5
-0.4
-0.3
-0.2
-0.1

banning thres

h
o
l
d

d

(
c
)

Figure 13: The download p erformance of free-riders and strategic p eers relative to the download p erformance of sharers using the banning p olicy for various thresholds – under (a) Central Max ow, (b) Distributed Max ow, and (c) Distributed Max ow Drop-Edge.
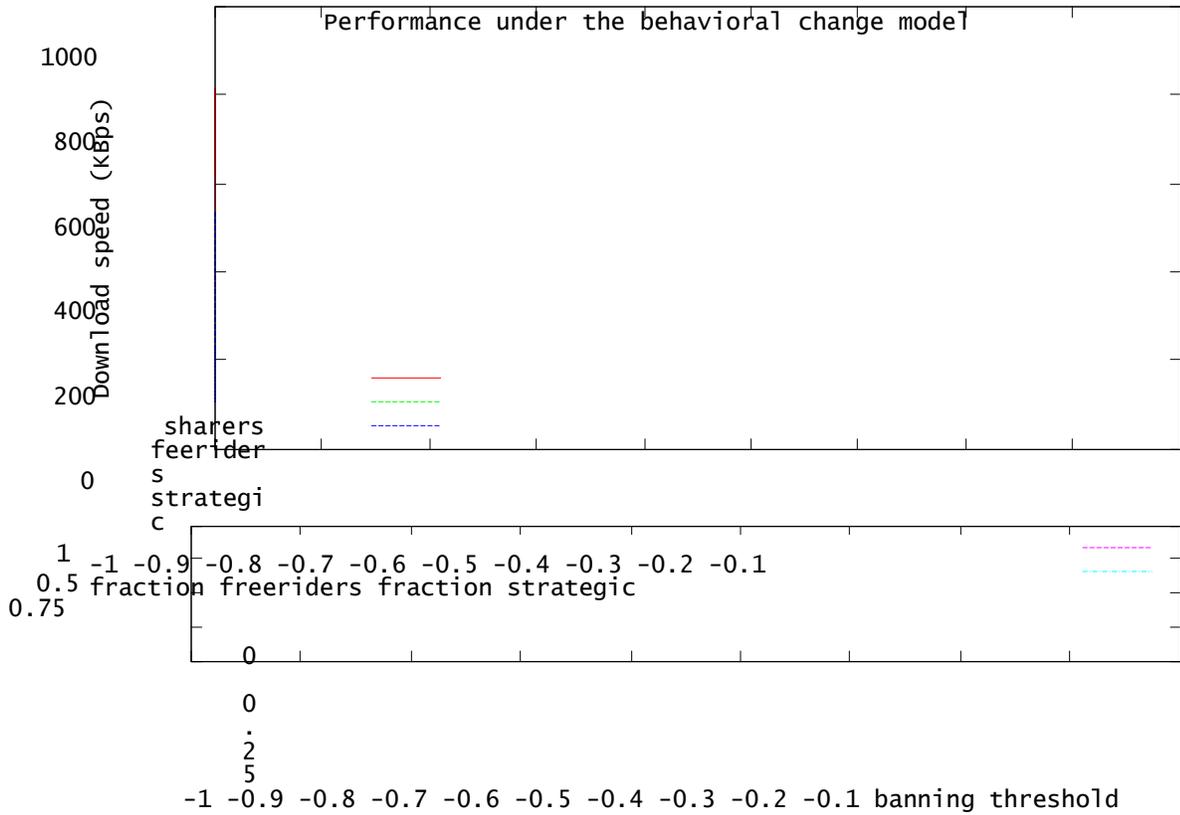27

Relative download performance

nce

Performance under the behavioral change model

1000

800

Download speed (kBps)

600

400

200

sharers
feeriders
strategic

0

1
0.5
0.75

-1 -0.9 -0.8 -0.7 -0.6 -0.5 -0.4 -0.3 -0.2 -0.1
fraction freeriders fraction strategic

0

0.25

-1 -0.9 -0.8 -0.7 -0.6 -0.5 -0.4 -0.3 -0.2 -0.1 banning threshold

Figure 14:

system designers can consider which of the two efiects is the most imp ortant. If one assumes that many free-riders will convert when they exp erience a severe p enalty, a strict threshold is b est, since in the end the overall system p erformance will improve for all p eers b ecause of the added resources of the 'ex-free-riders'. However, if free-rider conversion is slow, the prolonged loss of p erformance of the sharers might b e unacceptable, and a milder threshold should b e considered. In the next section, we will investigate this trade-ofi in more detail.

6.3.4 Behavioral change The goal of reducing the p erformance of free-riders is to motivate them to b ecome sharers. It is
reasonable to exp ect that in practice a system with a stronger p enalty for freeriding has less freeriders. The previous sections have shown that under the banning p olicy the p enalty for free-riders gets stronger when the p olicy is stricter, i.e., – closer to 0. Hence, we can exp ect that in reality there is some relationship b etween the setting of the threshold and the fraction of free-riders in the system. In order to gain an idea of the in uence of this relationship on the p erformance of sharers, free-riders, and strategic p eers, we p erformed simulations assuming the following b ehavioral relationship b etween the banning thresh-hold – and the p ercentage of free-riders in the systemf :

$f(-) = 0.8 \cdot (116)^{-}+1$; (19) With the ab ove relationship, a system with no banning (i.e., $-$ = −1) has 80% free-riders, while a system with very strict banning (i.e., $-$ = 0) has only 5% free-riders. As b efore, we assume that

28

in addition $0.25 \cdot f(\beta)$ of the peers are strategic. In Figure 14, we have displayed the download speed for sharers, free-riders, and strategic peers in a system where the above relationship holds. We observe that when the policy is very loose (β close to -1), all peers have a relatively low performance. This is intuitive, since because there is hardly any penalty for free-riders, many peers will freeride, which implies only little supply of resources. As the policy gets stricter, the performance of all peers naturally starts to increase. At some point ($\beta > -0.8$), there are apparently enough sharers in the system for the banning of free-riders to become effective. Around $\beta = -0.4$ we observe that the download speed of the sharers peaks, while the penalty for freeriding is very strong. As the policy gets even stricter ($\beta > -0.4$), we see that the banning of peers starts to have a negative effect on the performance of sharers. The explanation for this effect is that even a sharer might sometimes be banned because of the use of the distributed max-flow algorithm, i.e., some agents might falsely classify a sharer as a free-rider. Thus, the trade-off between sufficient banning of free-riders versus reducing unnecessary loss of performance for sharers is clearly visible in this figure. In practice, depending on the actual relationship f, it is up to community managers and system designers to devise policies that successfully balance this trade-off.

6.3.5 The Effect of Accounting Accuracy and Noise While the network size in our simulation is relatively small (100 agents), real BitTorrent networks
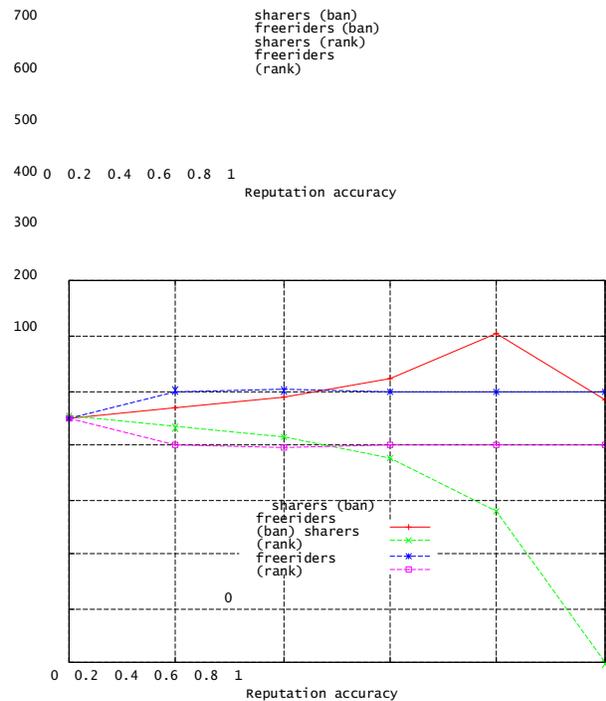are much larger, on the order of millions of agents.[8] In this section we study the effects of possibly disconnected graphs of agents and noisy information. In the following experiment, the agents do not compute the scores of the other agents explicitly, but instead we inject the scores, giving us the ability to control precisely how much and which information each agent has.

Fortunately, Piatek et al. [17] have shown that even in large networks, most agents are connected to each other via paths of length 2. However, not all agents will be connected to every other agent via such short paths. Given that in our max-flow implementation we only consider paths of length 3 or less, some agents will effectively have no information about some other agents in the networks. For the experiments we let the accounting accuracy denote the average percentage of agents that an agent has any information about. For example, an accuracy of 0.8 implies that on average, an agent is connected to 80% of the agents via paths of length 3 or less. Another effect that using max-flow causes is that the scores compute by max-flow are only approximations for the network performed by an agent. As we have shown in Section 4.1, on average, using max-flow will lead to the same relative scores of the agents. However, for individual agents' scores, using max-flow introduces some noise. In our experiments, the noise corresponds to the variance of the distribution from which we draw an agent's view of another agent's score.
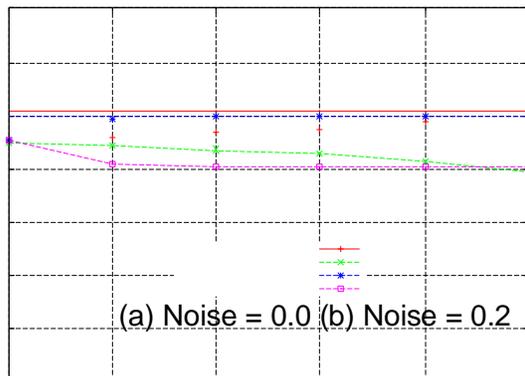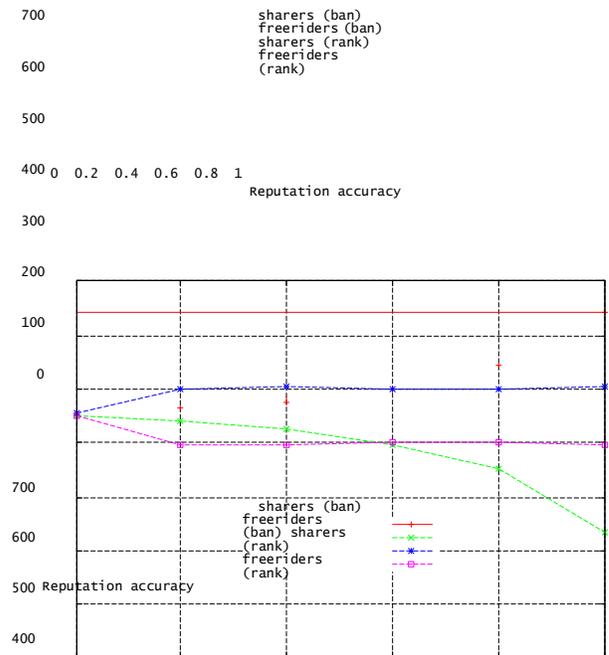
In Figure 15 we shown the results from these experiments. On the x-axis of all graphs, we vary the accounting accuracy between 0 (no information about any agent) to 1 (perfect information). The four graphs (a)-(d) correspond to noise values of 0.0 (no noise, i.e., perfect information), 0.2, 0.4, and 0.8. The mean of the distributions from which we draw the agent's scores were always the true scores. We also experimented with shifting the mean up or down (i.e., introducing systematic biases), but this did not lead to qualitatively different results.

The results for the ranking policy are very straightforward. Once the accuracy reaches the level of 0.2, the mechanism successfully separates sharers from free-riders, giving them a performance ration of 5 to 4, and this stays the same even as we increase the accuracy to 1.0. We have already explained
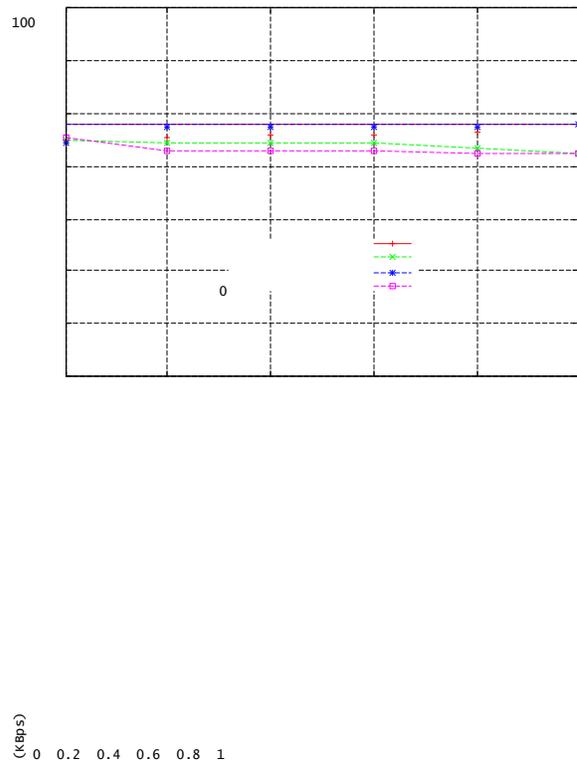
[8] We cannot simulate significantly larger BitTorrent networks on the protocol level because we need to compute each agent's score from each other agent's perspective, and this simply takes too long once we go beyond a certain network size.

700

600

500

400 0   0.2 0.4 0.6 0.8 1

300

Reputation accuracy

200

100

sharers (ban)
freeriders
(ban) sharers
(rank)
freeriders
(rank)

0

0   0.2 0.4 0.6 0.8 1

Reputation accuracy

30

sharers (ban)
freeriders (ban)
sharers (rank)
freeriders
(rank)

700

600

500

400 0   0.2 0.4 0.6 0.8 1

300

Reputation accuracy

200

100

0

700

600

500 Reputation accuracy

400

300

200

100

sharers (ban)
freeriders
(ban) sharers
(rank)
freeriders
(rank)

700

600

500

400

300

200

100
0

(a) Noise = 0.0 (b) Noise = 0.2

0

(c) Noise = 0.8 (d) Noise = 2.0 Figure 15: Analyzing the efiect of accounting accuracy and noise.

Average download performance (KBps)

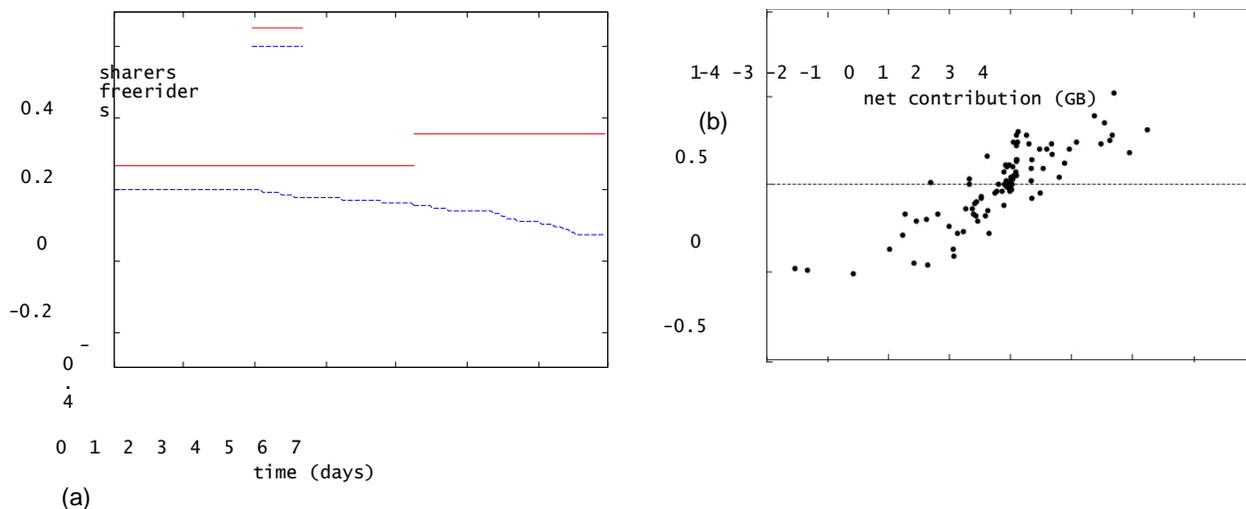Average download performance (KBps)

0   0.2 0.4 0.6 0.8 1

in Section 6.3.2 what the origin of this efiect is. By comparing graphs (a) through (d) we also see that intro ducing noise into the system has no efiect on the p erformance of the ranking p olicy, even up to a noise level of 0.8 (Figure 15 (c). The explanation is simple: the average score of the sharers is still much higher than the average score of the free-riders. Thus, even with some noise in the system, it is still very likely that the highest ranked agent (out of all agents) in any given choice set will b e a sharer and not a free-rider. Only once we increase the noise to 2.0 do es the result change a little bit, i.e., the p erformance for sharers decreases slightly and the p erformance of the free-riders increases slightly. However, such high values of noise are not realistic.

Now we turn our attention to the banning p olicy, and here the efiects of accounting accuracy and noise are much more pronounced. As we see in all graphs of Figure 15, the p erformance difierence b etween sharers and free-riders keeps increasing as we increase the accounting accuracy from 0 to 1. This is in fact exp ected for the banning p olicy, b ecause every agent that another agent has no information ab out will b e treated as an agent with a default score, and thus will not b e banned. Finally, let's consider the efiect of adding noise when using the banning p olicy. Here we see the biggest efiects: even going from no noise to a noise level of 0.2, the p erformance difierence b etween sharers and free-riders decreases signiflcantly. For noise level 0.8, the p erformance difierence achieved via the banning p olicy is almost consistently smaller than the one achieved via the ranking p olicy, except for very high accuracy values, where the two p olicies p erform essentially equally well. Again, this b ehavior is exp ected. By adding noise to the system, not only are some of the free-riders not banned that should b e banned, but also some of the sharers are now banned that should not b e banned. And the frequency of these mistakes is much higher than for the ranking p olicy, b ecause it applies to every agent that is considered for any upload slot, not just the agents b eing considered for the optimistic unchoking slot.

Now, to conclude this analysis we have to make some estimates regarding what accuracy and noise levels we can exp ect in real BitTorrent systems. As we have shown in our exp eriments b efore, even using the max- ow algorithm, the mechanisms compute scores that are highly correlated with agent's true net work. Thus, in practice, if an agent has information ab out another agent, the noise level can b e exp ected to b e relatively low. Furthermore, based on the results by [17], and assuming that we use a max- ow algorithm with a maximum path length of 3, an accounting accuracy of 0.99 can b e exp ected in real BitTorrent networks. Thus, we should consider Figure 15 (b) and lo ok towards the right end of the graph where we see that the banning p olicy causes a very large p erformance difierence for sharers and free-riders, which is the efiect we desired. Note that all of the results in Figure 15 are for one flxed value for the banning threshold. As we have explained in the previous section, by flne-tuning the banning threshold, we can p otentially further increase the p erformance difierence b etween sharers and free-riders.

6.4 Trace-based simulations

We also p erformed simulations based on network traces from the p opular private BitTorrent tracker filelist.org. The traces contain detailed b ehavior of all p eers that were active in the flle-sharing network, including uptimes, downtimes, connectability, and flle-requests. We have chosen this tracker b ecause it is not p ossible to scrap e such detailed information from most other trackers. The flle sizes in the trace range from several tens of megabytes to ab out one to two gigabytes, representing mostly audio and movie flles. As the traces do not contain information ab out the precise download pro cess and flnish time of downloads, we could use only the starting time in our simulations. We randomly divided the p eers into 50% sharers and 50% free-riders, and simulated the b ehavior describ ed ab ove. In our exp eriments, we compare the computed score of the p eers with their real net contribution.
31

Figure 16: (a) The system score in the system for b oth sharers and free-riders; (b) A scatter plot of the system score versus the net contribution of p eers (each dot represents one p eer).

Also, we compare the banning and the ranking p olicy in terms of download p erformance.

6.4.1 Contribution versus score We compare the real b ehavior of a p eer with its average score in the system. In order to measure a
p eer's real b ehavior, we compute a p eer's net contribution which is deflned as its real total upload minus its real total download during the simulation p erio d.

Figure 16 (a) shows a plot of the average system score of b oth the free-riders and the sharers throughout the simulation p erio d of one week. It is visible that the scores quickly diverge and that free-riders are clearly distinguished from sharers. In Figure 16 (b) the correlation b etween the net contribution and the system score of individual p eers is plotted, and the strongly p ositive correlation is apparent. Thus, despite the fact that p eers have incomplete information ab out other p eers, a p eer's resulting score in the system is a go o d representation of its real b ehavior.

6.4.2 Results for the banning p olicy To validate our results from the Poisson-based simulations, we have assessed the banning p olicy using
trace-based simulations. Figure 17 shows the average download sp eed of the sharers and the freeriders throughout the simulation p erio d. First of all, we observe that during the flrst days the average sp eed of free-riders is higher than that of sharers. This is b ecause free-riders do not allo cate any upload bandwidth to the sharing of complete flles and therefore have more upload capacity available to do tit-for-tat in their downloads. After some time however, the sp eed of the sharers increases and the sp eed of the free-riders decreases under b oth p olicies, leading to a signiflcantly higher sp eed for the sharers. At the end of the simulation p erio d, the free-riders have only ab out 50% of the sp eed of the sharers if the banning p olicy is used. Thus, we replicated the results from the Poisson-based simulations, showing that the accounting mechanism successfully separates sharers from free-riders.

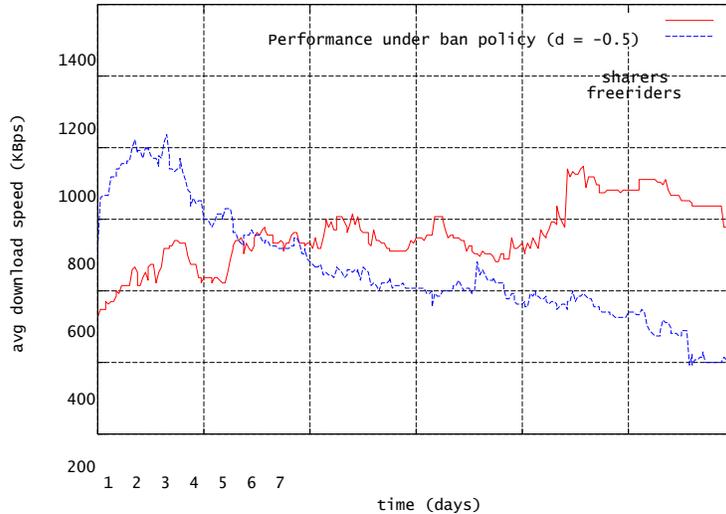average system reputation

reputation

Figure 17: The average download sp eed using the banning p olicy with – equal to -0.5.

7 Conclusion

In this pap er, we have formalized the problem of designing incentive-compatible accounting mechanisms for distributed work systems. We have shown that the existing BarterCast mechanism is highly susceptible to misrep ort attacks and that strategic agents could b eneflt signiflcantly from manipulating. To address this, we have intro duced the Drop-Edge extension which is misrep ortpro of. In our theoretical analysis we have proved that Drop-Edge provides go o d approximations to an agent's net contributions and scores resp ectively, and is accurate as the numb er of agents grows. However, we have also shown that neither BarterCast nor BarterCast+Drop-Edge is robust against sybil manipulations. This is no coincidence, as we have shown that there exists no sybil-pro of accounting mechanism that is resp onsive even to single agents' rep orts. We exaplain that the design of sybil-pro of accounting mechanisms would require relaxing this requirement and that sybil-pro of mechanisms would have to discard a signiflcant amount of information. Thus, in practice, a system designer would have to decide how imp ortant sybil-pro ofness is on the one hand vs. informativeness of the accounting mechanism on the other hand.

Via discrete simulations, where agents p erform discrete units of work in each round, we have shown that the go o d theoretical approximation ratio of the BarterCast+Drop-Edge mechanism also translates into go o d system e–ciency. When strategic agents are present, the Drop-Edge Mechanism clearly outp erforms BarterCast-basic: co op erative agents have higher e–ciency, while freeriding agents have lower e–ciency and we have shown that the magnitude of this efiect even grows over time. To test the efiectiveness of accounting mechanisms under more realistic conditions, we have implemented various accounting mechanisms into Tribler, a real P2P flle-sharing client that is already deployed and b eing used by thousands of users. Using Tribler, we were able to run simulations at the BitTorrent proto col level and thus obtain much more realistic exp erimental results. First, and most imp ortantly, we have found that the requirement of b eing backwards-compatible with existing BitTorrent clients puts some additional constraints on which accounting mechanisms we can usefully employ on top of a BitTorrent proto col. We have discussed in detail that using the ranking p olicy to make allo cation decisions has limited p ower in BitTorrent, b ecause it can at most achieve at 5 to 4 ratio regarding the p erformance of sharers vs. free-riders. The banning p olicy, in con-

33

trast, is more p owerful, and with a correctly-tuned banning threshold, it can separate sharers from free-riders such that sharers' p erformance is more than twice as high as that of free-riders. Under such conditions, it is realistic to assume that a signiflcant fraction of free-riders would change their b ehavior and b ecome sharers. Assuming such a b ehavioral change, we were able to show that using BarterCast+Drop-Edge we can achieve system e–ciencies signiflcantly higher than with the standard BitTorrent proto col. We have also provided a detailed analysis of the efiects of accounting accuracy and noise on the accounting mechanisms. It is necessary that the accounting accuracy is relatively high and noise levels are relatively low for the banning p olicy to separate the p erformance of sharers and free-riders to a large enough degree. Fortunately, based on previous results and our own exp eriments, we concluded that in real P2P flle-sharing networks, we can exp ect relatively high accounting accuracy and low enough noise levels, such that accounting mechanisms can b e exp ected to b e successful, even in very large networks with millions of agents.

References

[1] E. Adar and B.A. Hub erman. Free Riding on Gnutella. First Monday, 5(10):2, 2000. [2] Noga Alon, Felix Fischer, Ariel D. Pro caccia, and Moshe Tennenholtz. Sum of us: Strategypro of
    selection from the selectors. In Working Paper, 2010. [3] Nazareno Andrade, Miranda Mowbray, Aliandro Lima, Gustavo Wagner, and Matei Rip eanu.
    In uences on co op eration in bittorrent communities. In P2PECON '05: Proceedings of the 2005 ACM SIGCOMM workshop on Economics of peer-to-peer systems, pages 111{115, New York, NY, USA, 2005. ACM. ISBN 1-59593-026-4. doi: http://doi.acm.org/10.1145/1080192.1080198.

[4] Tina Balke, Marina De Vos, Julian Padget, and Frank Fitzek. Using a normative framework to explore the prototyping of wireless grids. In Proceedings of the Workshop on Coordination, Organization, Institutions and Norms in Multi-Agent Systems at AAMAS, 2010.

[5] Alice Cheng and Eric Friedman. Manipulability of PageRank under Sybil Strategies. In Proceedings of hte First Workshop of Networked Systems (NetEcon06), 2006.

[6] Bram Cohen. Incentives build robustness in bittorrent. In Proceedings of the Workshop on Economics of Peer-to-Peer Systems (P2PEcon), Berkeley, CA, June 2003.

[7] Michal Feldman, Kevin Lai, Ion Stoica, and John Chuang. Robust incentive techniques for p eer-to-p eer networks. In Proceedings of Electronic Commerce (EC), 2004.

[8] Michal Feldman, Christos Papadimitriou, John Chuang, and Ion Stoica. Free-Riding and Whitewashing in Peer-to-Peer Systems. IEEE Journal on Selected Areas in Communications, 24:5: 1010{1019, May 2006.

[9] Eric Friedman, Paul Resnick, and Rahul Sami. Algorithmic Game Theory, chapter ManipulationResistant Reputation Systems, pages 677{698. Cambridge University Press, 2007.

[10] Eric J. Friedman, Joseph Y. Halp ern, and Ian Kash. E–ciency and Nash Equilibria in a Scrip System for P2P Networks. In Proceedings of the 7th ACM Conference on Electronic Commerce (EC), pages 140{149, Ann Arb or, Michigan, 2006.
34

[11] M. Gupta, P. Judge, and M. Ammar. A reputation system for p eer-to-p eer networks. In Proc. of NOSSDAV'03, June 2003.

[12] Sepandar Kamvar, Mario Schlosser, and Hector Garcia-Molina. The EigenTrust Algorithm for Reputation Management in P2P Networks. In Twelfth International World Wide Web Conference, pages 640{651, 2003.

[13] R. Krishnan, M. Smith, Z. Tang, and R. Telang. The virtual commons: why free-riding can b e tolerated in p eer-to-p eer networks. In Workshop on Information Systems and Economics, Decemb er 2003.

[14] Qiao Lian, Yu Peng, Mao Yang, Zheng Zhang, Yafei Dai, and Xiaoming Li. Robust incentives via multi-level tit-for-tat. In Proc. 5th International Workshop on Peer-to-Peer systems (IPTPS), 2006.

[15] Thomas Lo cher, Patrick Mo or, Stefan Schmid, and Roger Wattenhofer. Free riding in bittorrent is cheap. In Proc. of HotNets-V, 2006.

[16] Michel Meulp older, Johan Pouwelse, Dick Ep ema, and Henk Sips. Bartercast: A practical approach to prevent lazy freeriding in p2p networks. In Proceedings of the 6th International Workshop on Hot Topics in Peer-to-Peer Systems (Hot-P2P), Rome, Italy, May 2009.

[17] Michael Piatek, Tomas Isdal, Arvind Krishnamurthy, and Thomas Anderson. One hop reputations for p eer to p eer flle sharing workloads. In Proceedings of the 5th USENIX Symposium on Networked Systems Design and Implementation (NSDI), pages 1{14, San Francisco, California, April 2008.

[18] J.A. Pouwelse, P. Garbacki, D.H.J. Ep ema, and H.J. Sips. The Bittorrent P2P File-Sharing System: Measurements and Analysis. In 4th International Workshop on Peer-to-Peer Systems (IPTPS), 2005.

[19] J.A. Pouwelse, P. Garbacki, J. Wang, A. Bakker, J. Yang, A. Iosup, D.H.J. Ep ema, M. Reinders, M.R. van Steen, and H.J. Sips. Tribler: A so cial-based p eer-to-p eer system. Concurrency and Computation: Practice and Experience, 19, 2008.

[20] Paul Resnick and Rahul Sami. Sybilpro of transitive trust proto cols. In Proceedings of Electronic Commerce (EC), 2009.

[21] Stefan Saroiu, P. Krishna Gummadi, and Steven D. Gribble. A measurement study of p eer-top eer flle sharing systems. In Proc. of Multimedia Computing and Networking 2002 (MMCN '02), 2002.

[22] Sven Seuken, Jie Tang, and David C. Parkes. Accounting mechanisms for distributed work systems. In Proceedings of the Conference on Artiflcial Intel ligence (AAAI), Atlanta, GA, July 2010.

[23] Jie Tang, Sven Seuken, and David C. Parkes. Hybrid transitive trust mechanisms. In Proceedings of the 9th International Conference on Autonomous Agents and Multi-Agent Systems, Toronto, CA, May 2010.

[24] V. Vishnumurthy, S. Chandrakumar, and E.G. Sirer. Karma: A secure economic framework for p eer-to-p eer resource sharing. In Workshop on Economics of Peer-to-Peer Systems, 2003.
35