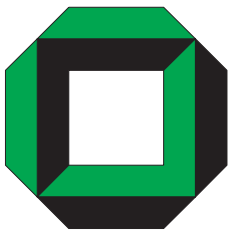
The Buddy System – A Distributed Reputation System Based On Social Structure

Stefan Fähnrich Philipp Obreiter

stefan@faehnrich.de, obreiter@ipd.uni-karlsruhe.de

March 16, 2004

Technical Report Nr. 2004-1



University of Karlsruhe
Faculty of Informatics
Institute for Program Structures and Data Organization
D-76128 Karlsruhe, Germany

Abstract

In P2P networks, there are no incentives to cooperate. There is neither a reward for cooperation nor a punishment for non-cooperation. A distributed reputation system could solve this problem, by giving means of managing trust towards other entities and discovering vicious entities. The existing distributed reputation systems are based on plausibility considerations and, thus, have several limitations. Therefore, in this report, we aim at overcoming these limitations by proposing the Buddy System as a distributed reputation system that is based on social structure. For this purpose, we discuss the design space of social structures and choose an appropriate social structure for the Buddy System. We consider implementation issues for its social structure by taking into account the volatility of the P2P network. Finally, we show by the means of simulation that the Buddy System significantly improves the effectiveness and efficiency of conventional distributed reputation systems. More specifically, the Buddy System is more effective in the detection of vicious entities and does not introduce any additional communication overhead.

Contents

1 Introduction

Peer-to-Peer (P2P) networks like KaZaA[1], Morpheus[2] or eDonkey[3] are widely spread and frequently in use throughout the internet[4]. Yet, all P2P systems are barely more than simple file exchange programs for almost anonymous partners. The degree of cooperation relies on voluntary participation and inoffensive behavior. No incentives or punishments are involved to ensure these two crucial necessities of P2P networks.

A plausibility based distributed reputation system solves the above problems, by giving means to rate other entities and share ratings with others. Hence, entities who try to exploit the network can be identified and ignored for future transactions. A distributed reputation system improves the degree of cooperation and furthermore ensures that the P2P networks is still functional. With an increasing number of vicious entities, a P2P network that lacks a distributed reputation system can easily perish. This is because well behaving entities have no motivation for participation if the degree of cooperation declines too much. Distributed reputation systems can be used in virtually any system of autonomous entities, such as P2P or ad hoc networks.

Still, distributed reputation systems are based on plausibility considerations and, thus, have to cope with inherent limitations. For example, self-recommendations are not possible, the impact of recommendation depends on one's own reputation, and only the recommender is in charge of the dissemination of trust values. Therefore, it seems necessary to introduce a paradigm that complements the plausibility considerations. For this purpose, we propose the use of social structure. The ensuing Buddy System combines the most suitable social structure and the functionality of plausibility based distributed reputation system. It improves the detection of vicious entities and, thus, increases the degree of cooperation.

This report is organized as follows: In Section 2, we introduce the system model of distributed reputation systems and state all assumptions for the Buddy System. Section 3 introduces social structures for distributed reputation systems and P2P networks. The design of the Buddy System is described in Section 4. The key algorithms for the social structuring are presented in Section 5 and the evaluation is shown in Section 6. Finally, we discuss the related work in Section 7 and conclude this work in Section 8.

2 System Model of Distributed Reputation Systems

The considered system consists of autonomous entities that may cooperate in the course of transactions. Each entity is autonomous and can therefore exhibit vicious behavior in any cooperation, i.e., it defects. Each entity runs an independent instance of the reputation system and reports any observed behavior to it. The instances of different entities may cooperate by exchanging recommendations. The system model which is used here is described in more detail in [1]. Figure 1 illustrates the model.

Assumptions. The following assumptions are made for the remainder of the report:

- Strong identities: No entity can change its identity or disguise as someone else (spoofing).

[1] www.kazaa.com
[2] www.morpheus.com
[3] www.edonkey.com
[4] http://www.sharmannetworks.com/content/view/full/255

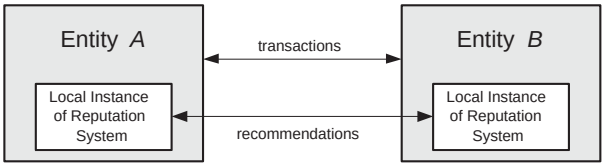


Figure 1: Model of a distributed reputation system

- Transactions are made in pairs.
- No observation of transactions between other entities: When a transaction is done, only the two participating entities know about the progress and outcome of that transaction. No other entity can overhear the outcome. Overhearing the outcome of a transaction is referred to as observation.
- Uncooperative behavior on lower protocol layers [2] is prevented.
- Confidential channels for communication exist: Confidential channels are important in P2P and ad hoc networks, since a message is typically routed along other entities. By making use of confidential channels, the entities that forward the message cannot overhear or change their content.

The restrictions that transactions can only be made in pairs and that no observation of other transactions is possible, make the detection of defection more difficult since no third party is involved in any kind.

Volatility. P2P networks have to deal with the volatility of the system. Entities can join and unlink at any time. No rules define how long an entity has to stay in the network or how often it can join. Therefore, the total number of entities participating – at any time – is unpredictable. Neither is, which entities are participating or for how long. These major fluctuations of the network have to be taken into account for a distributed reputation system and especially for its social structure.

Trust and reputation. Trust and reputation must be defined in the context of a reputation system. We are using the trust definition of Gambetta [3]:

> Trust (or, symmetrically, distrust) is a particular level of the subjective probability with which an agent will perform a particular action, both before [we] can monitor such action (or independently of his capacity of ever to be able to monitor it) and in a context in which it affects [our] own action.

In the context of reputation systems, we interpret the term *subjective probability* as being only the subjective probability of one agent, involving only personal experiences. Reputation shall be mainly defined as in [4]:

> The Reputation [...] is the average trust of all other entities [...] reputation clearly has a global aspect whereas trust is viewed from a local and subjective angle.

We have to adjust some parts of this definition for the use in distributed reputation systems. The term *of all other* is obviously not applicable, therefore it has to be interpreted as all other – *until then available* – entities. This restriction decreases the *global aspect* of reputation to a partly global aspect.

3 Social Structures for Distributed Reputation Systems

In real life, social structures are formed in order to achieve higher goals which would be impossible for the individual to achieve, such as preventing crime or building a skyscraper. To uphold the system, laws are enforced to ensure the rights of the individuals. For network communities whose users have a high level of anonymity, special considerations have to be made. Furthermore, some psychological and social findings should be discussed since they are applicable to P2P and ad hoc networks. In the following, we provide an introduction to this topic. Subsequently, we point out the specific issues that are relevant for ad hoc networks and, more generally, to P2P networks. Finally, we discuss effects of social structuring on distributed reputation systems.

3.1 An Introduction to Social Structure

The formation of groups has always been a crucial advantage for the human being. Because of our ancestral need to belong, we are a group-bound species. The psychologists Turner and Hogg pointed out that we identify ourself with certain social groups to state our individuality [5, 6]. For example, we identify ourself as a student, a men, a catholic and so on. Within a group we favor *ingroup* members and are sceptic towards strangers (the *outgroup*) [7]. We tend to cooperate within a group even if no reward is offered, just to strengthen the affiliation towards the group.

It seems reasonable to introduce social structure to P2P networks as well in order to strengthen the affiliation towards the network. The obvious problem is the users' anonymity in P2P networks. The higher the degree of anonymity the easier it is to overcome our guilty conscience. In an experiment of the New York University, it was shown that masked women used twice as much electric shocks than did identifiable women [8]. Therefore, it seems doubtful that entities will comply to group rules or even cooperate within a group. In contrast, the ingroup bias - the tendency to favor one's own group - holds even for an arbitrarily formed group. None of the so formed group members knew each other and they did not have the chance to get to know each other better. Still, when asked, they favored members of their "own" group [9, 10].

Group formation in P2P networks is a new area to investigate. Surely, none of these findings can be directly applied to P2P networks. Still, in internet communities with almost anonym members, the ingroup bias seems applicable as well. Therefore, there is a high probability that the findings are, in some way, transferable to P2P networks as well.

3.2 Social Structures

For the discussion of social structures, it seems reasonable to identify their key characteristics first. The most important aspect is whether all entities are *equal* concerning their trustworthiness or whether there exist dedicated entities that are inherently trustworthy and available. If such entities exist, the exclusion of vicious entities from future transactions is quite straightforward.

As soon as a *dedicated* entity exposes a vicious entity, it can forward the information to all other entities, who then refrain from any cooperation with that entity. In the absence of dedicated entities, the exposure of a vicious entity becomes more complex since no absolute identification

can be done. The success of forwarding the identification of a vicious entity depends on the level of trust the receiver has for oneself. In the following, we take a closer look at social structures that lack dedicated entities.

We distinguish the following characteristics of social structures: group size, entry and permanent requirements of groups, symmetry of grouping, reason of grouping, and stimulus of grouping. Note that all of the characteristics are orthogonal degrees of freedom for the design of social structures. This means that they are fully variable with each other.

Group size. Basically, there are two types of groups that are induced by the group size. *Bilateral groups* are groups formed by two entities. No other entity can join this bilateral group. *Multilateral groups* allow any number of members greater than two.

The larger the group is the harder is its maintenance. On the other hand, the more potential transaction partners are part of the group. Apparently, these two properties have to be traded off for the definition of the group size.

Entry requirements of groups. Entry requirements define under which circumstances an entity is allowed to enter a group. We distinguish between between structural requirements and group-specific requirements. *Group-specific requirements* are individually chosen by each group. Therefore, we cannot provide any general statement for such entry requirements.

Structural requirements define whether a member of a group is allowed to be a member of another group as well (*joint grouping*) or not (*disjoint grouping*). An example for disjoint grouping consists of a group that is formed for the development of software and does not want any member to join any other software development group. As for joint grouping, we can imagine a group that is formed to exchange garfield cartoons and has no interest to disallow their members to join any other group.

Permanent requirements of groups. The permanent requirements of a group specify the constraints that are valid for the group members. In general, the entry requirements are subsumed by the group rules. In analogy to group-specific entry requirements, *group-specific requirements* are individually chosen by each group. *Inter-group requirements* specify whether a group stipulates that its group members exhibit some kind of behavior towards members of other groups. For example, the software development group will probably be indifferent if a developer behaves viciously outside the group. If, on the other hand, the group is concerned that the reputation of the project could be at risk, they might demand that the developer exhibits good behavior.

Symmetry of grouping. We distinguish between symmetric and asymmetric grouping. *Symmetric grouping* refers to the term that the involved parties mutually agree or disagree to engage in some sort of grouping. In contrast, *asymmetric grouping* does not demand for such coordination of the involved parties. For example, this can be achieved by storing locally a list of trustworthy entities without disseminating it. Symmetric grouping enforces a global view on the network, while asymmetric grouping implicates a rather local view.

Reason of grouping. The main purpose of the formation of a group can be divided into two categories: A group might aim at achieving a *collective goal* of the group, as it is true for the software development group. Alternatively, the main purpose of the grouping is to support the

group members in achieving their *individual goals*. For example, the individual utilities of the group members could be increased by fostering cooperation among the group members.

Stimulus of grouping. The grouping may be defined exogenously or adaptively. *Exogenous grouping* ensues from a predetermined relationship between the entities. In such a case, the entry and permanent requirements are not orthogonal to this degree of freedom since they are predetermined as well. *Adaptive grouping* is performed according to some criteria of the social system. In this work, we are focussed on adaptive grouping since it allows the design of such criteria.

3.3 The Importance of Social Structure for Ad Hoc Networks

Most approaches only focus on how to identify vicious entities and not on how to attract well behaving ones. Furthermore, another main aspect has to be how anonymity can be overcome in order to give an incentive to cooperate. Overcoming anonymity is extremely crucial for volatile networks like ad hoc networks because a group feeling encourages people to join the network. We take the real life analogy of an airport. The more crowded with unknown persons a place is, the higher the need to belong and stick to a group [7].

The introduction of social structures to ad hoc networks is does not only improve the detection rate of vicious entities. In addition, the human nature is exploited in a way as well. Because of the psychological need to belong, people are given an incentive to cooperate ("to do it for the group").

Despite of the psychological advantages, there are the technical advantages as well. The degree of cooperation is increased through social structures. Furthermore, the detection rate of vicious entities and higher goals can be achieved through social structures by combining resources.

User mobility does not impose any problems to a social structure. However, the partitioning of the network might be a problem depending on the kind of group formations. If large groups have to split because of the partitioning, the group cannot accept new members or exclude old ones. This is because, for large groups, some kind of voting algorithm has to be processed in order to do so. Note that bilateral grouping does not have those problems.

3.4 Interrelation of Social Structures and P2P Networks

Social structures provide means for improving degree of cooperation in P2P networks. If social structures are formed, the members of a relationship may mutually grant privileges in some way. Those privileges can be getting more resources from other members, being favored for transaction demands or being actively warned about misbehaving entities. The degree of cooperation within the overall system is increased, as vicious entities will be excluded. On the other hand, this punishment is an incentive for members to cooperate. In order to join relationships, a high level of trust is necessary, which can only be gained by successful transactions. Therefore, an incentive for cooperation is given for newcomers.

Another reason why the degree of cooperation is increased is that members of a relationship warn each other about misbehaving entities. If that entity is not a member of the relationship, probably all other community member will refrain from making any more transactions with that entity. Therefore, a vicious entity cannot exploit as many entities as in a P2P network without social structure. Furthermore, the system has to prevent members of a relationship to defect against non-members since the reputation of the members would be at risk.

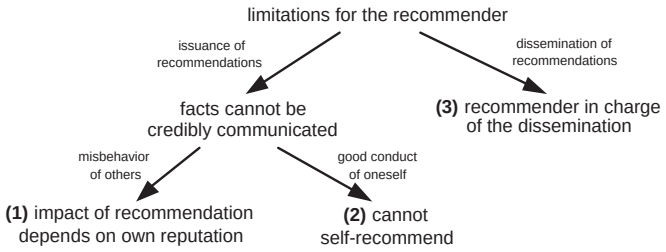


Figure 2: Limitations of existing reputation systems

With social structure, it becomes more difficult for a vicious entity and even for a coalition of vicious entities to exploit the system. A vicious entity by itself is less trusted than a member of a relationship. In order to enter into a relationship, the vicious entity has to cooperate first, therefore providing services for others. If that entity misbehaves after entering into a relationship, it gets excluded. In such a case, it adds more use to the P2P network then it can extract. Vicious entities can form together a coalition in order to skip the initial cooperation part. Yet, since the whole coalition is blamed for a defected transaction, all other vicious entities get less trustworthy by just one transaction. Hence, it is harder for vicious entities to exploit other entities.

All these aspects focus more on the technical side of a social structure. The psychological nature of a social structure probably has a high impact on the user as well. When a user is part of a relationship, he feels involved and committed to the other members of the relationship. Therefore, he has more reasons to cooperate and support other members, i.e., by staying longer online.

3.5 Social Structures and Distributed Reputation Systems

As mentioned in the introduction, distributed reputation systems have to cope with inherent limitations. We focus only on those that can be overcome by social structures. Figure 2 shows some of the limitations that are mentioned and discussed in [1].

In distributed reputation systems, the individual trust levels are passed on in order to inform other entities about personal experiences made. The entity which states the recommendation is called *recommender* and the receiving entity is called *recommendee*.

One of the limitations is that self-recommendations are not meaningful since no entity would spread negative information about oneself. With an underlying social structure, an entity can pass on the information in which relationships it participates. Together with the information about the number of relationship members and the entry requirements, an entity can convincingly state its trustworthiness. This information can easily be verified by asking the other members of the relationship.

Another limitation is that, for distributed reputation systems, the impact of a recommendation depends on the recommender's reputation and its plausibility. With an underlying social structure, the information about the recommender's membership in relationships can improve the trustworthiness of the recommendation. Therefore, the affiliation to a relationship can have a large impact on a recommendation.

The last limitation discussed here is that solely the recommender is in charge of a dissemination. Since this involves spending resources, it is not preferable for an entity to recommend frequently. This limitation can be overcome by stating that one of the relationship duties is that

recommendations have to be passed on at certain time intervals or when significant changes have been made.

The limitations of existing distributed reputation systems are summarized as follows:

- Self-recommendations are not possible/meaningful.
- The impact of recommendations depends on the reputation of the respective recommender.
- Recommender is in charge of disseminating the recommendation.

4 Design of the Buddy System

The Buddy System is a contextualized distributed reputation system with a social structure. In the following, we discuss its contextualization, its standard recommendation mechanism and its social recommendation mechanism. The contextualization facilitates the use of the Buddy System for different transaction contexts. The standard recommendation mechanisms capture the underlying distributed reputation system and how trust is stored and distributed. Finally, we choose a social structure for the Buddy System.

4.1 Contextualization

For any reputation system which is not specialized on one exact purpose, contextualization is crucial. In real life we trust our dentist and our mechanic, but we would never expect our dentist to fix a car.

The Buddy System distinguishes contexts regarding transaction value. For each transaction, an entity determines the transaction value and reports the outcome of the transaction and its value to the local instance of the reputation system. Therefore, for each transaction partner and each transaction value, there exists a trust value. It supports the choice of trustworthy transaction partners.

Each entity has full control over the contextualization and can combine different transaction values in groups as it wishes. The only restriction is that transaction value groups have to be disjoint in order to enable reasonable exchange of recommendations. Individual contextualization is extremely important for ad hoc networks where entities with different kind of devices cooperate.

Entities with different contexts can exchange their trust values by combining recursively two transaction contexts into a new one. This is performed by taking the minimum trust value of the underlying contexts. If the transaction context does not match a request, we choose the next higher transaction context which fully includes the requested transaction context. By taking the minimum value, a pessimistic approach is realized. Since the recommender has no exact information about the requested context, he is not sure how the recommended entity will behave in the new context. The recommender can only assume that it will correlate to one of the underlying contexts. In order to avert overestimation, the minimum trust value is chosen. If the approach was optimistic, gaining trust in one context could be sufficient to achieve trust in all higher transaction contexts.

Figure 3 shows an example for this. At the lowest level, the transaction contexts of an entity A are illustrated with the corresponding trust values for another entity B. In the context from 0 to 15, a trust of 2 is stored. If another entity C requests a recommendation, for example in context [0,12), then this is a subcontext of [0,15) and entity A can submit the corresponding trust of 2. If C requested a recommendation in the context [0,18) then none of the lowest level contexts of A

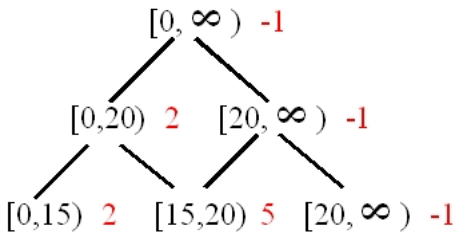


Figure 3: Exchange of Contexts

would match. Hence, entity A creates new contexts by combining the original context. Thereby, the contexts [0,15) and [15,20) are combined into a new one, with the minimum trust of the underlying trust values. Finally, entity A can submit the corresponding trust value to entity C.

4.2 Standard Recommendation Mechanism

4.2.1 Trust

Trust is stored and calculated in a straightforward manner. After each transaction, the outcome and context of the transaction is reported to the local instance of the reputation system. It calculates the new trust value for the entity. For this purpose, the following rules apply: After a defected transaction an entity is downgraded. If the transaction was successful, increasing the trust value depends on the current trust value and all recently successfully completed transaction with this entity. For an upgrade $2^{trustvalue}$ successfully completed transactions are needed. If this is the case, the number of the transactions is decreased by that value and the trust value is increased by one. For an upgrade from trust value 2 to trust value 3 a total of four successful transactions is needed. For example entity A has a trust level of 3 of entity B. The successfully completed transaction counter is set to 7. After the next transaction, the trust for Entity B is increased to 4 and the counter is set back to zero. For the next upgrade 16 transactions are needed.

4.2.2 Recommendations

We distinguish between direct and indirect recommendations. Direct recommendations only include first hand experiences of the recommender. In contrast, for indirect recommendations, the recommender also considers recommendations of other entities. For the Buddy System, we only make use of direct recommendations.

Hybrid recommendations are a mixture of direct and indirect recommendation. For the calculation of personal trust, available recommendations are taken into account. Therefore, the trust value does not offer a distinction between first hand experiences and recommendations. If this trust value is shared then this is a hybrid recommendation.

Direct recommendations. Direct recommendations are done in a straightforward manner. Recommendations are sent on demand or after transactions. The receiving entity can weight those recommendations according to the personal trust level towards the recommender. The recommender trust is stored separately for all recommending entities. By this means, the recommender trust can be recalculated if personal trust values change.

Indirect recommendations. Indirect recommendations have multiple disadvantages: mostly the communication and computation overhead is increased dramatically. For indirect recommendations, more requirements must be met. Every recommendation has to be non repudiable in order to avoid changing the content. Further, every recommendation has to be attached with a timestamp, because multiple recommendations from the same entity can exist in the network. In order to reply to a recommendation request of an entity correctly, the personal trust value as long as all indirect recommendations would have to be passed on. This would increase the cost of a single message to $O(n)$ (n=number of entities). Apart from the higher communication overhead, the storage capacity of an entity has to be increased as well. This is a crucial aspect, since in an P2P network probably not all entities have enough storage to manage indirect recommendations.

Moreover, the enhancement of indirect recommendations for the network is doubtful. Obviously the number of recommendations an entity can obtain in a short time is increased, but the recommendee can only evaluate those recommendation of the entities it knows from personal experience. Recommendations of unknown entities should not be examined, since first the entity itself would have to be reviewed. For that direct communication would be necessary, and therefore an indirect recommendation would not make sense.

For all of the above reasons indirect recommendations are not supported by the Buddy System.

Hybrid recommendations. Hybrid recommendations are difficult to take into account. This is because the trust value does not offer a distinction between personal and recommended trust and the receiver does not know the weightings the recommender used. Furthermore, indirect recommendations can spread uncontrollably if the recipient of a hybrid recommendation adjusts his trust assessment and issues a further hybrid recommendation. This means that a recommendation about entity A can be taken into account twice: First, directly by the trust values of B and C towards A. Second, if B recommends A to C the recommendation is taken another time indirectly into account. Therefore, the Buddy System does not make use of hybrid recommendations.

4.3 Social Recommendation Mechanism

In the following, the social structure of the Buddy System is presented and the advantages of the specific buddy structure are pointed out.

Choice of the social structure. The Buddy System is a distributed reputation system for equal entities. In the following, the aspects introduced in Section 3.2 are discussed for the Buddy System. Later in this section, each aspect is discussed in more detail

- **Group size**: only bilateral groups
- **Entry requirements of groups**:
  - a minimum level of mutual trust
  - similar "views" of the network (see Section 5.1)
  - none with regard to the number of group memberships (joint grouping)
- **Permanent requirements of groups**:
  - Buddies have to favor each other.
  - Buddies have to verify self recommendation checks.

  - After certain time intervals, the buddy relationship has to be verified again.

- **Symmetry of grouping**:
  - The buddy relationships are symmetric

- **Reason of grouping**:
  - **Individual goals:** Increase of the member's individual utility by mutual cooperation
  - **Collective goals:** Overcoming the limitations of conventional distributed reputation systems.

- **Stimulus of grouping**:
  - The stimulus of a buddy relationship is adaptive, depending on the specific needs of the entity.

By this quite simple buddy structure, several limitations of distributed reputation systems can be overcome. Self-recommendations are possible by stating the number of buddies the recommender has. Obviously the more buddies one entity has, the trustworthier it appears since it has already proven to be trustworthy to those entities. With an increasing number of buddies, the impact of an entity's recommendation is increased as well because this entity has proven trustworthy in the past.

In conclusion, the volatile nature of the buddy relationships matches the nature of P2P networks.

Management of the social structure. The number of buddies that one entity can have is not limited. A buddy relationship is formed when both entities agree to do so. It can be ended at any time by one of the entities without specific reason given. An annulment of a buddy relationship is realized as soon as one entity requests the annulment and the other entity confirms the request.

Problems can arise if the partner buddy refuses the annulment simply by "playing dead". In this case, it is difficult to reconstruct what happened. The partner buddy might have deliberately refused to send an acknowledgement or it is simply out of reach or its device is turned off. Different approaches can be pursued:

1. **Immediate cancellation:** The buddy relationship is cancelled without any acknowledgement. Tradeoff: the other entity might end up as a liar.

2. **Lazy cancellation:** The buddy relationship is cancelled when the buddies enter into their next transaction.

3. **Timeouts:** After a certain time interval the buddy relationship is cancelled if not confirmed otherwise.

4. **Third-party mediation:** On next buddy examination of another entity, confirm request, but demand other entity to pass on the annulment request.

New problems arise from the latter approach. If still no acknowledgement is obtained, we are in doubt which entity defected. Nevertheless, the Buddy System uses a combination of lazy cancellation and third-party mediation to cope the problems of annulment. The further

implications of third-party mediation are ignored since it is assumed that the problems will not arise frequently. Lazy cancellation alone is sufficient if both buddies are within reach since it is assumed that buddies are often participating in transactions. Third-party mediation is added to cope with partitioning of the network. If a buddy examination reaches an entity, it knows that the network is either reunited (then we can still uses lazy cancellation) or that the requester will soon be able to reach the buddy. The latter can only be the case if the requester is highly mobile and moves between two partitions. Since this case will not arise frequently, we use an optimistic approach when using third-party mediation.

For self-recommendations, the number of buddies is stated in order to indicate one's own trustworthiness. Therefore, checking the accuracy of the stated number of buddies is crucial. For such check, the supposed buddies are requested for the confirmation of their relationship with the self-recommender. Alternatively, such check may be probabilistic by randomly checking just a few buddies. In the Buddy System, the following approach is realized: A rather small number of buddies are verified in order to keep the communication overhead small. For the evaluation of the recommendation, three outcomes are possible: 1) all requested buddies confirm so that the total number of stated buddies is used, 2) some buddies were not available, so the total number of stated buddies is reduced by the percentage of those not answering compared to those who confirmed, 3) at least one buddy declines the relationship with the self-recommender so that the number of buddies is set to zero and the recommender is informed.

The permanently evolving Buddy System is harder to overview than any other social structure since every entity only knows a very specific part of the system. In order to get an overview, the buddies of every entity would have to be processed into a table. This is very costly and time intense. Hence, the Buddy System refrains from doing so. The other way round, this is another advantage of the Buddy System since, for a vicious entity, it is hardly possible to identify the actual structuring and advance against certain groups of entities.

Recommendations in the social structure. Recommendations are issued on request. Therefore, getting recommendation is no longer solely in hands of the recommender. Note that no active warning of defectious entities is achieved. This is done out of two reasons: First the communication overhead is reduced and second each entity can choose its own level of risk averseness. Active warning mechanisms could be abused in the fashion of an denial-of-service attack. Furthermore, an entity might not be interested in being warned, e.g., because the warning's context appears to be irrelevant for its specific needs. Nevertheless, a risk-averse entity may still obtain many recommendations by actively requesting them.

5 Social Algorithms in the Buddy System

In this section, we present two algorithms that are used for the social structuring in the Buddy System.

5.1 Choice of Buddies

procedure $checkForBuddy(possiblePartner)$

1: $allPeople \leftarrow getAllKnownPeople()$
2: **while** (allPeople.hasMoreElements()) **do**
3: $\quad currentPerson \leftarrow allPeople.nextElement()$
4: $\quad myTrust \leftarrow getTrust(currentPerson)$

```
    hisTrust ← getRecommendTrustFrom(possiblePartner, currentPerson)
    if (myTrust ≠ 0 & hisTrust ≠ 0) then
      diff ← calculateDifferences(myTrust, hisTrust)
      distance ← distance + diff
      users ← users + 1
    else
      Ignore
    end if
    if checkForDifference(myTrust, hisTrust) then
      difference ← difference + 1
    end if
  end while
  if ( (distance < config1) & (users > config2) & (difference > config3) ) then
    RETURNtrue
  else
    RETURNfalse
  end if
```

This algorithm describes what is verified for a buddy relationship. Before the procedure is called the trust value for the requester is verified. For all known entities (trust value $\neq 0$) the trust values are compared. All differences are added up ($distance$), the total number of compared entities is stored ($users$) and the total number of differences ($differences$). If all requirements are met, the request is accepted. The verifying of differences is used because buddies should have equal or almost equal views of the network to avoid later conflicts. The $config1-3$ values can be set individually depending on the risk awareness of the entity.

5.2 Choice of Transaction Partners

The following algorithm is not a reputation system issue. Rather, it illustrates how the Buddy System can be used to support the transaction decision process.

```
  procedure getBestPartner()
  allPartners ← GetAllKnownPartners()
  while (allPartners.hasMoreElements()) do
    currentPartner ← allPartners.nextElement()
    calculateRecommendTrust(currentPartner)
    checkIfBuddy(currentPartner)
  end while
  sortedList ← sortPartnersByTrustLevels(allPartners)
  returnRandomElement(sortedList)
```

This straightforward algorithm shows how transaction partners are chosen. A list of all entities is created, sorted by trust values. Buddies receive an additional bonus. Then randomly, one of those entities are chosen for a transaction. If that entity is not available, it is deleted from the list and the next one is chosen. The probability of being chosen declines logarithmic. Therefore, the most trusted entity is chosen with a probability of 50%, the second one with probability 25% and so on... This algorithm is also applied if there is a queue for requested transactions. By this means, the rights and duties of buddies are enforced as well since buddies are sorted at the top of the list and are therefore preferred. The probability that one of the first 6 entities is chosen is over 98%, which seems unreasonable regarding the possibly large network. As long as the entities

are available and cooperate, this makes perfect sense because, for an entity, the outcome of a transaction is most important. If an entity defects (possibly unvoluntarily), the trust value is decreased and the probability of being chosen declines. Then, other entities get the chance to prove their trustworthiness.

6 Evaluation of the Buddy System

In this section, we take a closer look at the evaluation of the Buddy System. This includes a brief discussion of the implementation issues. Based on the implementation, the test scenarios and test results are presented.

6.1 Implementation issues

The implementation of the Buddy System was done in the context of the DIANE Project of the University of Karlsruhe. For the implementation of the Buddy System, the following decisions have been made:

- The scale of trust values was set from -7 to 8.
- A trust value of 0 indicates an unknown entity.
- The trust values -7 and 8 indicate static trust or distrust.
- $\sum_{i=0}^{n} abs(x_i - y_i) < k$ and a minimum trust of 4 must be fulfilled to enter a buddy-relationship.
- The rate for buddy review upon self-recommendation was set to 10%.
- Transaction contexts were only value based. No semantic contexts were used.
- Recommendational trust is stored as decimal place.

The trust scale is set from -7 to 8, whereby 0 indicates unknown entities and -7 receptively 8 denotes static trust. Static Trust [2] refers to the idea that an entity can set the trust level of another entity to a unchangeable value. Hereby, an entity which knows another entity from real life could set the trust to 8. The reputation system cannot change that value. Only the entity itself can set the value back to a dynamic value, which is changeable by the reputation system.

$\sum_{i=0}^{n} abs(x_i - y_i) < k$ is the formula for buddy-relationships. x and y denote the two entities that are requesting for a buddy-relationship. x_i is the trust level entity x has about entity i, abs denotes that the absolute value is taken into account. k refers to the minimum distance that the buddy requestor must have in order to enter an relationship. This depends obviously on all known entities and is set by default on all entities divided by two. The algorithms of Section 5.1 is applied for this purpose.

6.2 Simulation Runs

We have used the benchmark group *IBR2*[5]. The instances of the benchmark group differ in the network's volatility and the ratio between cooperative and vicious entities. Vicious entities always

[5]The detailed description of the benchmark group IBR2 is found in the Appendix A.

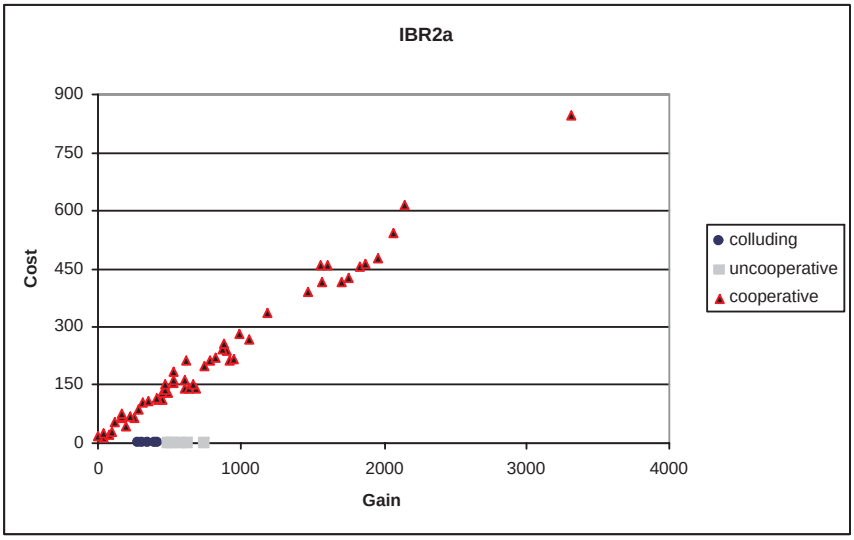


Figure 4: Individual gains and costs for the Buddy System (IBR2a benchmark)

defect in the course of transactions, whereas cooperative entities only enter into transactions with rather trustworthy entities. There are two types of vicious entities, i.e., uncooperative and colluding entities. In contrast to the uncooperative entities, colluding entities make active use of the Buddy System by mutually forming buddy relationships.

The simulation has been run three times with each of the five IBR2 instances. First, without any reputation system. Second, with the Buddy System that had its social structure inactivated. Therefore, it was merely a regular distributed reputation system (DRS). The third run was performed for the Buddy System that had its standard recommendation mechanism inactivated. By this means, we are able to compare the effectiveness and efficiency of standard and social recommendation mechanisms.

Figure 4 illustrates how the test runs were examined. For each user the individual gains and costs out of all transactions was determined. As the figure shows, uncooperative and colluding users did not have any costs since they always defect in the course of transactions. The costs and gains of the cooperative users varied depending on how long they stayed online. Even though the uncooperative and colluding entities stay online the entire time, their gains are less than those of cooperative entities.

This becomes even clearer if we subtract the entities which were only online for a short time. The uncooperative entities are discovered very effectively since they cannot pull out much profit. This is shown in Figure 5. At the end of each simulation run, the correlation of the individual costs and gains is calculated. The ensuing coefficient of correlation indicates the degree of linear relationship of individual gains and costs. Furthermore, a positive correlation shows that the gain increases with rising costs, whereas a negative correlation implies that vicious entities can gain more than cooperative ones. We conclude that the coefficient of correlation measures the fairness of the reputation system[6]. In the following, we focus on such coefficient in order to obtain an aggregated view of the simulation results.

[6]A more detailed analysis of this means of evaluation can be found in our previous work [11].

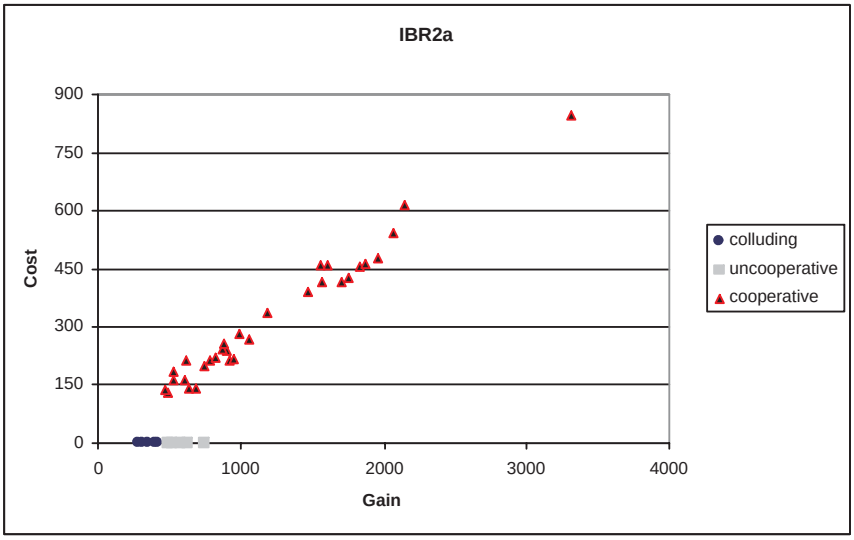


Figure 5: Revised individual gains and costs for the Buddy System (IBR2a benchmark)

Colluders. As Figure 4 shows, colluding users are less efficient in exploiting the network then regular uncooperative ones. This finding has to be explained in more detail since it represents a crucial advantage of the Buddy System. Colluding entities participate in buddy relationships but they do not make transactions with each other. Therefore, they try to exploit the self-recommendation mechanism of the Buddy System. When a new entity enters the network, the colluders can self recommend and therefore seem trustworthy for the newcomer. At first, colluders attract newcomers. Nevertheless, this is an acceptable tradeoff since after one colluder defects the whole group is downgraded. Therefore, colluders can be efficiently discovered. This is essential to the robustness of the Buddy System.

Correlation for different benchmarks. Figure 6 gives an overview of the effectiveness of the Buddy System. Apparently, without any distributed reputation system (DRS), correlation and therefore the effectiveness is at a very low level. The introduction of the standard recommendation mechanism of the Buddy System yields an improvement of the correlation. It is further increased by the application of the social recommendation mechanism of the Buddy System. Even though the improvement does not seem significant at first sight, it is quite remarkable. Since the standard recommendation mechanism already are at the top 80-90%, further improvement is exceptional. On the average, the social recommendation mechanism improves the effectiveness of the standard recommendations by another 15-20%.

In benchmark IBR2c, which consists of 15 uncooperative, 15 colluding and only 10 cooperative users, the effectiveness of the social recommendation mechanism outperforms the standard recommendation mechanism by far. Still, there is no high correlation since there are too many vicious entities.

Newcomers. The success of newcomers is another very important aspect of how efficient a reputation system is. In the initial phase where no entity knows any other, it is easier for vicious entities to exploit cooperative users. After transactions occurred among most of the entities,

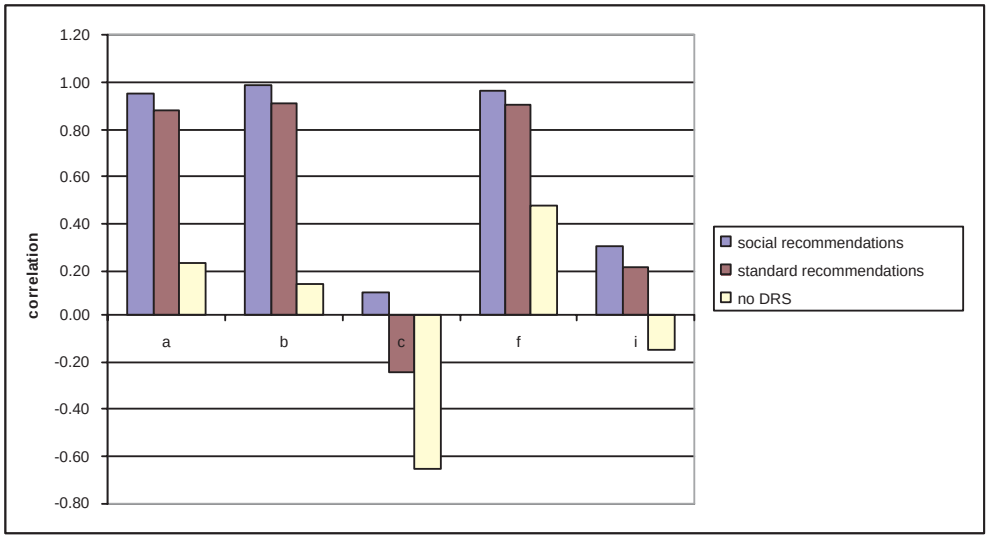


Figure 6: Coefficient of correlation for selective instances of the IBR2 benchmark group

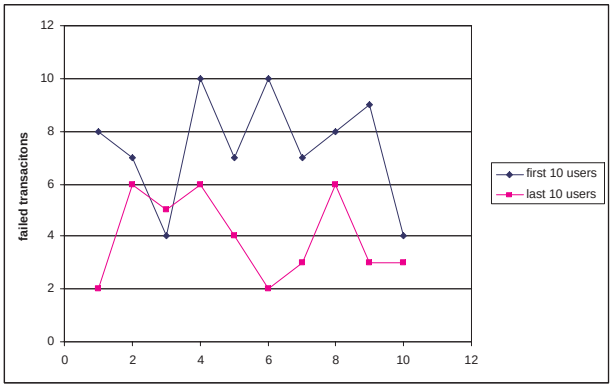


Figure 7: The number of unsuccessful transactions for newcomers (IBR2a benchmark)

newcomers should be warned efficiently. The warning of newcomers can only be achieved with an underlying social structure. Otherwise, recommendations would not be meaningful since the newcomers do not know how to weight the recommendations. In the Buddy System, cooperative entities can self-recommend by stating the number of their buddies, whereas uncooperative users cannot do so since they do not participate in buddy relationships. Therefore, it is more likely that newcomers choose cooperative users than uncooperative users.

Figure 7 shows how effective the warning of newcomers is for the IBR2a benchmark. At the beginning, most cooperative entities have not yet been able to form buddy relationships. Therefore, no significant self-recommendation can be done. At the end of the simulation run, the recommendations are more meaningful so that the overall failed transaction ratio drops. At the beginning, an average newcomer is involved in 7.4 unsuccessful transactions. At the end of the run, this number is decreased to 4. This is especially remarkable since transaction partners are randomly chosen and colluders are self-recommending as well. Because of the random factor, the

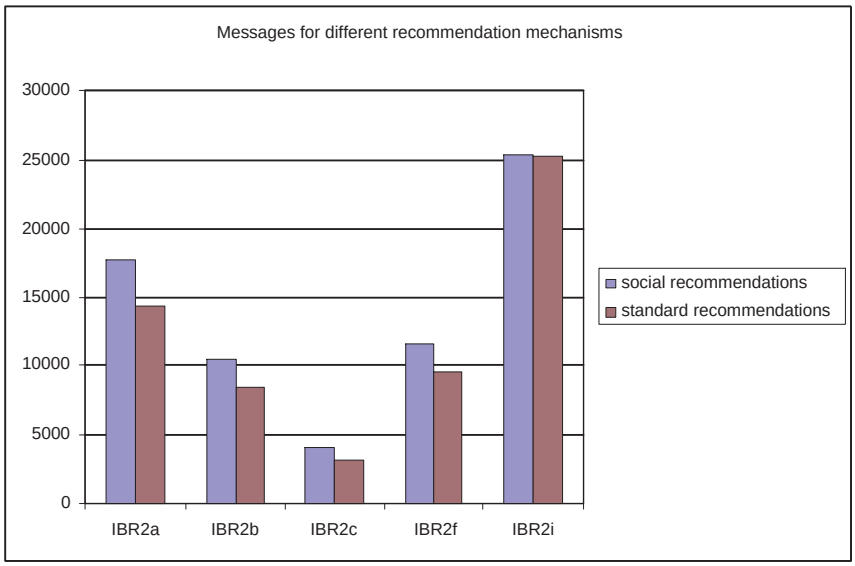


Figure 8: The number of overall messages for social and standard recommendation mechanisms

lines are not straight but fluctuate.

Messages. In order to assess the overhead of standard and social recommendation mechanisms, we compare the number of sent messages for different recommendation mechanisms in Figure 8. It shows that the number of overall messages of social recommendation mechanisms is about 10–15% higher. Note that, for standard recommendation mechanism, recommendations are sent either after a successful transaction or before entering into a transaction with a prior unknown entity. For the social recommendation mechanism of the Buddy System, recommendations are exchanged before entering into a buddy relationship and self-recommendations are requested when encountering a prior unknown entity.

Figure 9 breaks down the number of sent messages into three classes: recommendations, self-recommendations and maintenance messages. Recommendations are standard recommendations as used in any distributed reputation system. Note that the social recommendation mechanisms of the Buddy System cannot do completely without such recommendations since this is the most effective way to test whether the potential buddies' views of the world are likewise. Still, two entities only exchange recommendations if they are about to become buddies. In any other situation, self-recommendations are obtained in order to examine an entities trustworthiness. Maintenance messages are messages sent either when requesting someone to become a buddy or checking whether a claimed buddy relationship actually exists. Note that recommendation messages are hybrid in the way that they have a twofold purpose: Mainly, they enable the calculation of the buddy requirements. But since they have to be obtained anyway, the recommendational trust is calculated and stored as well. Figure 9 shows that self-recommendations are sent most often.

In Figure 10, we consider the size of the sent messages. For this purpose, we make the following assumptions: Every entity has on average one quarter of the system's entities as buddies. This assumption overestimates the size of buddy messages since such number of buddy relationships can be hardly achieved. This is because first enough trust towards the other entity must be

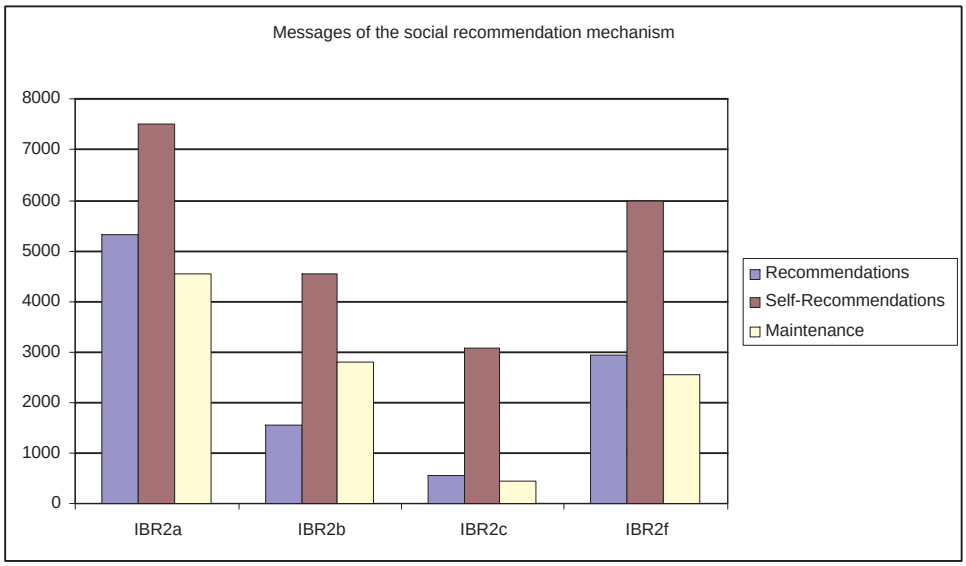


Figure 9: Messages of the social recommendation mechanism of the Buddy System

established, which assumes that the other entity is cooperative and a transaction has been made. In addition, the recommendation size is underestimated since it is assumed that the entity only knows half of all entities. A further assumption is that the identifier of an Entity consists of 20 bit, which would limit the overall network size to one million users. The size of a transaction context was evaluated to 11 bit which would represent a transaction value of maximum of 1024. Since an interval rather then a concrete value is requested we need the 11th bit. Further, the trust value needs 4 bits. Maintenance messages are very small in size since they only consist of one entity ID. Maintenance messages either checks whether an entity is a buddy or, if someone else wants to become a buddy, requests a standard recommendation for the computation of the similarity of world views. Therefore, the costs - in bits - of all messages are: (n represents the number of all entities)

- Self Recommendations: $n/4$ * 20
- Regular Recommendations: $n/2$ * (20 + 11 + 4)
- Maintenance Messages: 20

With these assumptions, we calculate the total costs. They are shown in Figure 10. The size is measured in KBytes. Apparently, the overall costs for the messages of the social recommendation mechanism are far less than for the standard recommendation mechanism. This is because standard recommendations are far more expensive. Note that, even aside the specific assumptions regarding the relative number of buddies, the size of a standard recommendation can be always assumed at a minimum of twice the size of a self-recommendation. This is because every entity knows obviously more entities than it has buddies. Furthermore, the mere size of a recommendation is a least 35 bits, while a self recommendation only need 20 bits.

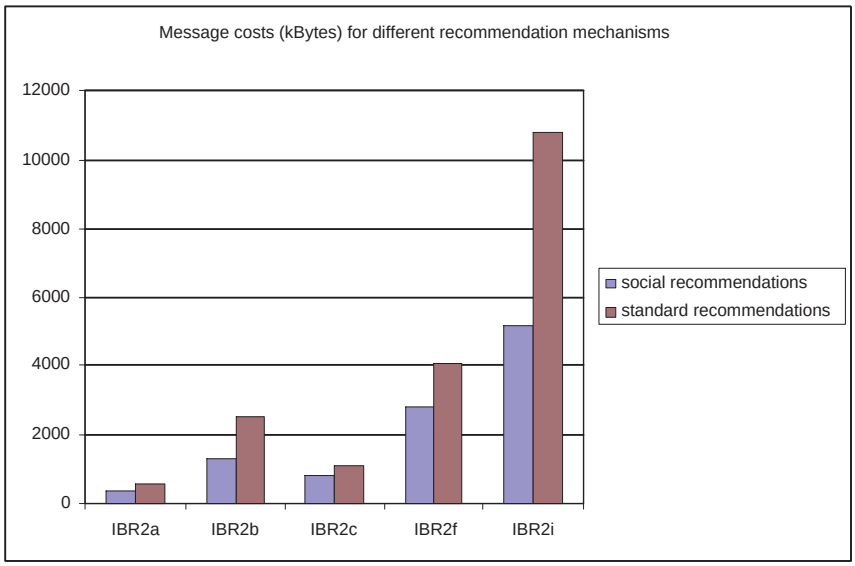


Figure 10: Message costs for social and standard recommendation mechanisms

Summary. The scenarios show that even the standard mechanisms of the reputation system significantly improve the degree of cooperation. The social structure of the Buddy System further enhances the effectiveness of the reputation system. It is crucial that the Buddy System has a high robustness. This means that the social structure can not be exploited by vicious entities. Furthermore, even since the total number of messages increases, the total costs for all messages decreases, since self recommendations are less expensive. Overall, the performance is increased by 15–20%, while the overall cost for messages is decreased.

7 Related Work

7.1 Existing Distributed Reputation Systems without Social Structures

Many approaches have been made designing and improving reputation systems [12, 13, 14, 4, 15, 2].

Probably one of the best known distributed reputation system is the CONFIDANT Protocol [13]. CONFIDANT is the abbreviation for "Cooperation Of Nodes, Fairness In Dynamic Ad-hoc NeTworks" and is used to detect misbehaving entities. An entity consist of a monitor, a reputation system, a trust manager and a path manager. Each entity can monitor the packages sent to and from neighboring entities. Thereby, entities can detect misbehaving entities, and report this to the reputation system. If a certain threshold is reached, the path manager is used to route packages around the misbehaving entity and ignore further requests. Further, an ALARM is triggered to inform others about the misbehaving entities. The receiver of an ALARM messages examines it through the trust manager to verify it's trustworthiness. Significant improvement in throughput can be achieved.

Another well known approach is the CORE (collaborative reputation mechanism) mechanism [15]. Similar to the Monitoring Unit of the CONFIDANT Protocol, a watchdog unit is used, to overhear the messages sent from other entities. The reputation system differentiates be-

tween personal experience, recommendations and a functional reputation value, which represents some sort of context aware value.

Different approaches have been made to improve distributed reputation systems [12, 14].

Dellarocas introduces a clustering approach to filter false praises [14]. Through the robust algorithm a high detection rate of colluding entities can be maintained. Buchegger uses a statistical approach to identify misbehaving entities [12]. The Bayesian algorithm uses the number of suspected misbehaving and well behaving entities as input for the beta function of a binomial equal distribution.

7.2 Existing Distributed Reputation Systems Based on Social Structures

Despite of all the approaches to improve distributed reputation system, some limitations can not be overcome by simple statistical means. These limitations are discussed in detail in [1] and have been partly examined in Section 4.

Miranda and Rodrigues designed a distributed reputation system for routing purposes [16]. Each entity has three variables: $friends$, the entities it cooperates with, $foes$, the entities is refuses service too and $selfish$, which includes entities which treat oneself as a foe. Theses variables are exchanged with other entities. Messages can be overheard, and if no messages are heard from an entity for some period of time all information will be deleted. This poses the major disadvantage of this approach, since identified foes will not be recognized when they reenter the network. Nevertheless, the friends-and-foes reputation system represents an interesting approach since it makes use of social structures. It applies *multilateral* groups which aim at achieving *individual goals*. The *entry requirements* are good behavior. There are no further *permanent requirements*. Note that the grouping issue is dealt in a personal rather than a global view. By this means, each entity denotes its own friends and foes (by subjective means), which can be asymmetric. This means that the other entity doesn't necessary have to think the same way. Since the friends and foes lists are distributed to others, the relationships should become eventually symmetric.

To our knowledge, the work of Buskens and Weesie [17] represents the only approach that explicitly considers both social structure and distributed reputation systems. For this purpose, the authors assume a social structure that is *bilateral*, *joint*, *exogenous*, and *asymmetric*. The *reason of grouping* is collective since it aims at the identification and isolation of untrustworthy entities. In order to facilitate a game theoretic analysis of such a setting, the authors make restricting assumption with respect to the system model: **(1)** There exists an entity (so-called trustee) that is proposed transactions with other entities (so-called trustors). **(2)** The issuance and reception of recommendations may only include the trustors of two subsequent transactions. In addition, a trustor only recommends probabilistically to the members of the groups that it belongs to. **(3)** Every recommendation is truthful. **(4)** The transaction peers have perfect information of their incentives to defect. Based on these assumptions, the authors analyze the impact of different social structures on the effectiveness of the reputation system. In this regard, they provide guidelines for the definition of exogenous social structures.

8 Conclusion

P2P networks are frequently in use, still the network relies solely on altruistic users. No incentives or punishments are given to enforce cooperation. A distributed reputation system combined with a social structure improves the P2P network and assures its mere existence. In this report, the

notion of distributed reputation systems has been introduced. An introduction to the psychological aspects of social structure has been given. Furthermore, the advantages of social structures for P2P networks have been pointed out. Emphasis has been laid on the design space of social structures and their adjustment to the volatile nature of a P2P network. We discussed the design of the Buddy System as a distributed reputation system that is based on social structure. We showed how it overcomes the limitations of conventional distributed reputation systems. By the means of simulation, we have shown that the Buddy System improves the degree of cooperation and therefore the overall quality of a P2P network. Furthermore, the robustness of the Buddy System has been shown.

In the future, we will examine the impact of non-repudiable tokens [1] on the Buddy System. By this means, the effectiveness of the Buddy System could be further improved.

Acknowledgement

The work done for this report has been partially funded by the German Research Community (DFG) in the context of the priority program (SPP) no. 1140.

Bibliography

[1] Obreiter, P.: A case for evidence-aware distributed reputation systems. In: Second International Conference on Trust Management (iTrust'04), Oxford, UK (2004)

[2] Obreiter, P., König-Ries, B., Klein, M.: Stimulating cooperative behavior of autonomous devices - an analysis of requirements and existing approaches. In: Proceedings of the Second International Workshop on Wireless Information Systems (WIS2003), Angers, France (2003) 71–82

[3] Gambetta, D.: Can we trust trust? In Gambetta, D., Blackwell, B., eds.: Trust: Making and Breaking Cooperative Relations, Oxford (1990) 213–237

[4] Kinateder, M., Rothermel, K.: Architecture and algorithms for a distributed reputation system. In: Proceedings of the First Intl. Conf. on Trust Management (iTrust), Heraklion, Crete, Greece (2003) 1–16

[5] Turner, J.C.: Rediscovering the social group: A self-categorization theory. Basil Blackwell, New York (1987)

[6] Hogg, M.A.: Intragroup processes, group structure and social identity. In Robinson, W., ed.: Social Groups and Identities: Developing the Legacy of Henri Taiffel, Butterworth Heinemann (1996)

[7] Myers, D.G.: Psychology. 6th edn. Worth Publishers (2001)

[8] Zimbardo, P.G.: The human choice: Individuation, reason, and order versus deindividuation, impulse, and chaos. In Arnold, W.J., Levine, D., eds.: 1969 Nebraska Symposium on Motivation, Lincoln, University of Nebraska Press (1970) 237–307

[9] Tajfel, H.: Social identity and intergroup relations. Cambridge University Press, New York (1982)

[10] Wilder, D.A.: Perceiving persons as a group: Categorization and intergroup relations. In Hamilton, D.L., ed.: Cognitive processes in stereotyping and intergroup behavior, Hillsdale, N.J., L. Erlbaum Associates (1981) 213–257

[11] Obreiter, P., König-Ries, B., Papadopoulos, G.: Engineering incentive schemes for ad hoc networks - a case study for the lanes overlay. In: First International Workshop on Pervasive Information Management (EDBT-Workshop), Heraklion, Greece (2004)

[12] Buchegger, S., Le Boudec, J.Y.: The effect of rumor spreading in reputation systems for mobile ad-hoc networks. In: Proceedings of WiOpt '03: Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks, Sophia-Antipolis, France (2003)

[13] Buchegger, S., Boudec, J.Y.L.: Performance Analysis of the CONFIDANT Protocol: Cooperation Of Nodes — Fairness In Distributed Ad-hoc NeTworks. In: Proc. of IEEE/ACM Workshop on Mobile Ad Hoc Networking and Computing (MobiHOC), Lausanne, Switzerland, IEEE (2002) 226–236

[14] Dellarocas, C.: Immunizing online reputation reporting systems against unfair ratings and discriminatory behavior. In: Proceedings of the ACM Conference on Electronic Commerce, Minneapolis, MN, USA (2000) 150–157

[15] Michiardi, P., Molva, R.: Core: A collaborative reputation mechanism to enforce node cooperation in mobile AD HOC networks. In: Proceedings of the 6th IFIP Communications and Multimedia Security Conference, Portorosz, Slovenia. (2002)

[16] Miranda, H., Rodrigues, L.: Friends and foes: Preventing selfishness in open mobile ad hoc networks. In: Proc. of the First Intl. Workshop on Mobile Distributed Computing (MDC'03), Providence, RI, USA (2003)

[17] Buskens, V., Weesie, J.: Cooperation via social networks. Analyse & Kritik **22** (2000) 44–74

Appendix A

Benchmarks

1 IBR2

This benchmark group is characterized by the following:

- Each entity runs the same components that are configured the same.
- The population is fixed at 50 devices that are mutually always reachable.
- Only cooperative, uncooperative, and colluding devices take part.
- Colluding devices form exactly one collusion. The members of a collusion mutually set static trust values in order to promote each other.
- The life-cycle of cooperative devices is as follows: They enter the network, participate in some transactions and finally leave the network. In order to keep the population fixed at 50 devices, a cooperative device enters the network whenever another cooperative device leaves.
- Transactions are performed according to the barter trade pattern. No exchange protocols are used. Furthermore, it is assumed that each entity has to decide whether to execute its action before it knows about the behavior of its transaction partner (PD-game). The transaction context is a general one.
- Each transaction peer is able to fully perceive whether the transaction partner defected or not. By this means, the perception is without noise.
- This benchmark group assumes that a user is asked whenever a bilateral transaction with an other entity would make sense. Only if both users agree to such a transaction, it is performed. For this purpose, each peer may define whether it defects in the course of that transaction. At last, each peer is informed of the outcome of the transaction. This means that a transaction represents a prisoner dilemma game.
- For each transaction, an entity can choose among three entities (=possible transaction partners).
- The set of possible transaction partners is chosen randomly. By this means, there is no predefined locality of cooperation behavior. Yet, the entities' choice of transaction partners should provide for locality.

- Every device of the population is always online.
- The gain-cost-ratio is 3.
- The entities are risk neutral.

Bench-mark	#coop Users	#uncoop Users	#collud. Users	#trans-actions	entrance rate
IBR2a	35	10	5	5000	100
IBR2b	35	0	15	5000	100
IBR2c	10	15	15	5000	100
IBR2f	35	10	5	10000	100
IBR2i	35	10	5	5000	30

Cooperative users, uncooperative users and colluding users: This benchmark group makes use of the transacting user. Cooperative users cooperate if the valuated gain is higher than the costs of the transaction (they never defect). Uncooperative ones always participate and defect in transactions. Colluding ones always participate and defect in transactions with entities that are not member of the collusion.

The choice of transaction partners is as follows: Cooperative users choose the most trusted entity, whereas uncooperative/colluding users choose randomly.

Overall transactions: It indicates the number of proposed transactions. The entity that is proposed a transaction is chosen pseudo-randomly.

Entrance rate: Specifies how many transactions are proposed before a cooperative user leaves and another enters.