Faculteit der Exacte Wetenschappen

Master Thesis

DRAFT: Discounting experience in referral networks

Author
Nicolas Höning

Supervisors
Martijn Schut
Gusz Eiben

August 14, 2009

"The two offices of memory are collection and distribution; by one, images are accumulated and by the other produced for use."

Samuel Johnson

"Forget the past – the future will give you plenty to worry about."

Goerge Allen

Abstract

In the context of information systems, a disruptive environment demands a solution to a trade-off: How quickly should agents forget experience? If they cherish their memories, they can build their decisions on larger data sets; if they forget quickly, they can respond well to change. This task can be characterised as a decentralised learning problem and its solution highly depends on the environment. In this work, we establish a testbed to examine this problem by building on a trust network model by Hang et al. [2008]. We observe which forgetting patterns work best and what happens if agents freely choose their rate of forgetting.

Contents

1. Introduction

Information becomes less accurate over time as conditions change in a dynamic environment. More recent information is valued higher - this is a side effect of forgetting, but no coincidence. It reflects uncertainty in a changing world.

In computer science, trust has become the preferred concept to model the transforming of experiences into opinions about one another. Referral systems, which define protocols to disseminate social information in trust reports, will be at the core of many electronical multi-agent systems. While well-designed multi-agent systems are not "Social Capital" (Putnam [2000]) by themselves, they are a modern "Social Technology" (Nelson [2003]), which might prove very helpful in organising people and communication more efficiently.

However, the notions of certainty and especially forgetting have not gotten a lot of scientific attention. This is unfortunate, since especially sources of misunderstandings in communication, for instance between two agents using different views on the past, deserve our consideration, so that our social interactions actually profit from social technology. Furthermore, we should be eager to learn about interactions between the micro- and macrolevel in referral systems. Between building up experience and reacting quickly to change, every agent makes the local decision how much information of the past to dismiss. The consequences on the global level have (to the authors best knowledge) not been subject of systematic research.

In this work, we attempt to establish the exchange of certainty-based trust reports among autonomous agents as a research design for the study of information networks and in particular pay attention to the role of discounting (forgetting). As starting point, this thesis takes a short paper with experiments conducted by Hang et al. [2008], who created a network simulation for certainty-based trust referrals. It will rebuild it, extend it and study various parameters for effects on system performance. Then we will examine the role of discounting more closely: How do differing discounting strategies affect system performance and what, if any, patterns of discounting strategies emerge if agents can choose their discounting strategy freely?

We build upon previous work by Audun Jøsang and Munindar P. Singh to model trust and its operators mathematically. In particular, we use the model outlined in the Phd thesis of Yonghong Wang, which incorporates the notion of uncertainty and is tuned to work well with conflicting information.

In particular, our work proceeds in three phases:

1. Consolidation We rebuild the experiments from Hang et al. [2008] and consider extensions to the model, such as discussions of in- and output modelling and the revision of the update operator.

2. Exploration We implement our disruptions/risk model and then conduct experiments to test for effects of (dis)honesty of agents.

3. Adaptation We add adaptive behaviour to the agents: First, referrers update their trust in one another and we test for effects of structural parameters and patterns of discounting strategies. Then, agents get to decide which discounting rate they should use and we look for emerging effects on system performance.

Tables 1.1, 1.2 and 1.3 list our experiments and their goals [1].

We proceed as follows: Chapter 2 describes relevant work in the domains of trust representation, certainty-based trust, referral networks and discounting. Chapter 3 introduces the objectives of this research and chapter 4 describes the model we use. In chapter 5, experiments an their results are described. In chapter 6, we conclude and discuss further research.

Nr	**Title**	**By**	**Goals**
1	Network Trust Approximation	Hang	Measure the ability of the network to estimate experience in uncertain settings.
2	Damping Referrer	Hang	Observe memory effects with different discounting factors when an agent changes behaviour.
3	Damping provider, honest referrer	Hang	Observe how trust in a referrer solely depends on his accuracy, not the kind of information.

Table 1.1.: Experiments in Consolidation Phase

[1] Some more experiments were used as integration tests and can be found in the Appendix.

Nr	Title	By	Goals
3b	Discounting Alignments	Höning	Show that it is hurting adaptability if client and witnesses use different discounting factors.
4a	Disruption/Risk-Model I	Höning	Introduce disruption/risk model and observe the adaptiveness of different discounting strategies.
4b	Disruption/Risk-Model II	Höning	Observe the system performance (the clients payoff) with different discounting strategies under worsening disruptiveness.
5	Populations with mixed honesty	Höning	Observe system performance with various mixes of honest/dishonest agents.

Table 1.2.: Experiments in Exploration Phase

Nr	Title	By	Goals
6	Adaptive Referrers	Höning	Give referrers feedback, so they can deal with local knowledge only, observe performance.
7	Structure	Höning	Try different connectivity settings and observe performance.
8	Discounting Distributions	Höning	Try different distributions of the same amount of discounting among the agents and observe performance.
9	Adaptive Discounting	Höning	Let agents adjust their discounting factor, observe resulting strategy pattern and performance.

Table 1.3.: Experiments in Adaptation Phase

2. Relevant Literature

Trust has been a thriving research area in Computer Science for about 15 years and still is (e.g. Castelfranchi and Falcone [1998], Jonker and Treur [1999], Khambatti et al. [2004]). In the following section, we will look at existing definitions of trust and related concepts.
The two main questions in trust research are how to model trust of one individual in another individual and how trust networks behave, where we also deal with reputation, the trust of others in others. While this research focuses on the latter question, both questions are inseparable as we hope to make clear later on in this chapter. We build our work on a mathematically sound, certainty-based trust representation for our model, which will be discussed in section 2.2. Referral networks will also be mentioned, but are then discussed in detail in section 2.3. We will finally look at techniques to address the trade-off between recency and certainty in section 2.4, and ground our research interest in this topic.

2.1. Trust and Reputation

2.1.1. Trust

In this section, we define trust as a probability measure towards others and reputation as a collective measure. We sketch the notion of reputation systems. Finally, we distinct three types of trust: personal trust, role-based trust and collective trust.

In Computer Science, the term *trust* is generally agreed upon to denote a subjective opinion which an agent holds about another agent (we will also call the two agents in a trust relation the *trustor* and *trustee*, respectively). Intuitively, trust is the most basic concept of Artificial Intelligence. To be able to reason about the world, any individual must model the agents in it. In its most simple form, trust is formalised in a bipolar way - a scalar $\in [0, 1]$ denoting bad trust, good trust or any degree in between.
The trust opinion can be about something specific, i.e. the ability of the trustee to provide a certain service. A trustor could hold several trusts about the same trustee - each describing his belief about a distinct ability of the trustee. Thus, for computer scientists, trust is the simplest concept of mind imaginable as it is often represented by a simple scalar and can be reused in a modular way to model more knowledge about the world.

We start by considering some definitions of trust and how they introduce notions deemed important for research on trust. A simple definition is delivered by Demolombe [2001]:

"We can understand trust as an attitude of an agent who believes that another agent has a given property."

Gambetta [1990b] adds that trust is almost never certain, but a measure of probability. Embedded in this definition is also how trustor and trustee have an asymmetric dependancy-relation:

"*Trust is the subjective probability by which an individual, A, expects that another individual, B, performs a given action on which its welfare depends.*"

Chervany and McKnight [1999] stress the dependancy of trustor to trustee even more, making it the central part - trust is how much the trustor is willing to depend:

"*Trust is the extent to which one party is willing to depend on something or somebody in a given situation with a feeling of relative security, even though negative consequences are possible.*"

Note how this definition also states how disappointment is always a possibility. It gives trustors the benefit of relative security, but in the end, to rely on trust entails risk. It is mostly agreed that a trust relation is greatly defined, even initially enabled, by the risk involved.

2.1.2. Reputation

When trust systems are discussed, there is often a confusion between trust and reputation. In contrast to trust, the reputation of an agent is what the system thinks about him. The Oxford Dictionary defines: "*Reputation is what is generally said or believed about a persons or things character or standing.*" Jøsang et al. [2007] write:

"*Reputation can be considered as a collective measure of trustworthiness (in the sense of reliability) based on the referrals or ratings from members in a community.*"

Thus, reputation is an abstraction over the trust opinions of many.
Research in biology has focused on reputation mechanisms to explain cooperation. Sigmund and Nowak [1998] formalised a process called Image Scoring, where each agent has a global score which increases when he cooperates and decreases when he defects. Milinski [2001] use a mechanism called Standing, in which an agent "*loses good standing by failing to help a recipient in good standing, whereas failing to help recipients who lack good standing does not damage the standing of a potential donor*".

2.1.3. Reputation systems

Sociologists, Economists and computer scientists share their main interest in reputation as a protocol in social systems, e.g. like in eBay. In these systems, agents can share trust reports with each other, in order to disseminate valuable information.
For instance, Resnick et al. [2006] conducted experiments to assess the economic value of reputation for a power-seller on eBay. Misztal [1996] discusses trust in social systems as a means of social cohesion. Her main three observations are that trust makes social life predictable, creates a sense of community, and it makes it easier for agents to cooperate. The first and third of these observations immediately appeal to computer scientists as they strive to model social contexts in a way that is useful for both humans and the machines they employ. For instance, consider the computer science research by Faehnrich and Nimis [2004], whose main observations are that reputation systems give orientation to newcomers, protect against collusion and disseminate information.
In contrast to many reputation scores (e.g. Image Scoring or Standing), the reputation any agent derives from the network are his local view and not global to the whole system.

Singh and Yolum [2003] established the term *referral network* for a social system that uses trust as a protocol and trust reports as a commodity to share among agents. In these systems, agents build referral paths, connecting them to to the target agent vie referrals. From such a paths, agents extract a reputation score. We note that this score resembles the combined trust opinion of the agents local view on the network and is not global. Furthermore, it is of interest to note how direct experience (trust) becomes indirect knowledge (reputation) when it is shared. In fact, each agent updates his own trust with the opinions referred to him by others and thus his opinion becomes a mix of trust and reputation. This is the reason that trust and reputation are not only hard to distinguish, they are inseparable.

By simulating these systems, their dynamics can be made visible and discussed. What resembles the social networks of humans best? What is efficient in terms of computation and fairness? To be more specific, how should agents integrate trust reports from different sources? How should they update their own reports when new information arrives? These questions can be discussed by formalising operators in a trust domain. In section 2.3 we will look at a specific set of them and also discuss referral networks in more detail. Lately, also security in trust networks has moved into the focus of reseach, e.g. in Kerr and Cohen [2009]. The strategies this thesis uses are rather simplistic, as we focus on the system behaviour (we explain the model in chapter 4).

2.1.4. Personal trust, role-based trust and collective trust

Originally, the concept of trust is inspired by observing human interactions. Sociologists have argued about the types of trust that are actually employed in society (e.g. Gambetta [1990a]). Most scholars describe two notions of trust - we will discuss this distinction as it clarifies the context of our work.
Luhmann [2000] differentiates between *trust* and *confidence*. In his view, trust is used

when an agent is personally engaged in a familiar one-to-one relation, while confidence is the attitude an agent develops towards the social system he reacts with.The trustor may interact with a single agent, but to him this agent represents the trustee, the organisation. For instance, to visit a doctor requires confidence in the medical system. When this doctor has become your personal doctor which you visited several times, you approach him with trust.
In the same way, JOHANSEN [2007] uses the terms *personal trust* and *role-based trust*, respectively. Johansen also uses the term *depersonalised trust* for role-based trust, which maybe decribes it best.
For our purposes, this distinction is especially helpful when considering which contexts computer science is actually most interested in. Those are modern, electronically interconnected social systems in which participants often engage with each other for the first time or in an unfamiliar context. Considering the informational progression of our social systems, Luhmann writes:

"*These new conditions, of access and temporal pressure, of opportunity and dependence, of openness and lack of integration, change the relation between confidence and trust. Trust remains vital in interpersonal relations, but participation in functional systems like the economy or politics is no longer a matter of personal relations. It requires confidence, but not trust.*"

We agree with Luhmann in this point. Trust, as understood as a complex and very personal matter of involvement, is not what is modeled in contemporary trust research in computer science (or artificial intelligence, for that matter). Mostly, these trust systems model Luhmanns confidence or Johansens role-based trust, especially since in most of them, trust is enriched with reputation, a collective view on trustworthiness. Simply put: As ones trust opinion gets influenced by the opinion of others, it becomes less personal. Yet, depersonalised trust is the state of the art in trust research - a sophisticated one-to-one relation between two machines or two programmes has yet to be build.
It is not implied here that artificial intelligence is not interested in modeling personal trust more detailed in the future. This will be worked on in a short future, but we need to learn more about the mechanisms behind it from new advances in (Neuro-)psychology (e.g. the role of the hormone Oxytocin in trust, in KOSFELD ET AL. [2005]) or other disciplines.
Luhmann uses a further criterion to distinguish trust from confidence. In both situations, the trustor faces the possibility of disappointment. According to Luhmann, confidence entails *danger*, which is independent from ones action and hard to predict (e.g. the general danger of being badly informed in a censorship society). Trust, on the other hand, involves *risk*, which is highly dependant on the trustors further action and often higher than the danger which comes with the confidence relation (e.g. the risk of exchanging regime-critical letters in a said society). In Luhmanns view, the distinction between danger and risk is an ability that societies can evolve:

"Whereas in the Bible, for instance, the Last Judgement comes as a surprise, the late Middle Ages began - under the influence of the confessional - to represent it as the predicted outcome of risky behaviour. In committing sins you risk the salvation of your soul, which thereby becomes a matter not of church practice but of individual lifestyle and effort."

Luhmann further explains how confidence and trust enable each other in the light of the danger/risk distinction. High-risk engagements come about in societies in whose subsystems the citizens have high confidence. For example, the more confidence you have in the law system, the more you may feel able to risk an up-front investment in a personal business relation [1].
The point of these considerations is that the confidence-building social systems which are currently in the focus of computer science research (and also of this thesis) are much needed buildings block for our society in that they help humans to stay connected, informed and confident. As has been mentioned before, these social systems are what Nelson [2003] calls "Social Technologies". Following Luhmann, they are a crucial ingredient for an effective society in which personal trust blooms.

However, the distinction between personal and role-based trust (or trust and confidence) does not describe fully the interconnected settings of referral networks. We noted before how role-based trust models the trustee as a (depersonalised) group or system. In contrast, a referral system models the trustor as a group. Essentially, the trustor makes use of a hive mind when he collects the reputation score. We think it is accurate to speak of collective trust (Nowostawski and Foukia [2007] also use this term), when a network opinion made from several trust values is produced. Collective trust closely resembles reputation, but captures the notion how, given time and connectedness of the trustee, collective trust values might differ substantially. Also, personal experiences might very well be mixed in.

[1] Luhmann also sees an influence of trust on confidence and border cases where confidence blends into trust or vice versa, but these details are out of the scope of this section

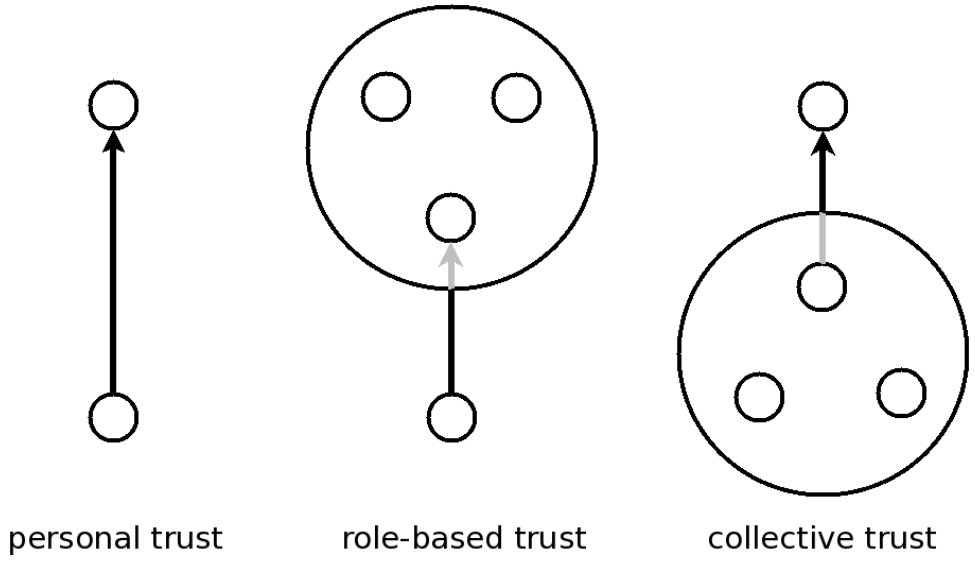


Figure 2.1.: A simple comparison of trust types

2.2. Certainty-based Trust

In this section, we introduce the certainty-based trust representation we use for our model. We explain its intention and research history. We will then formulate how belief models in this representation work in detail.

Trust is based on experience. The more experience a trustor collects about the trustee, the more accurate his opinion (his trust report) should become. For this, every trust representation needs to be accompanied with an update function, so that new experience somehow alters the existing opinion.
A trust report is thus nothing more than a compressed history of experience. When agents translate their experience history into their trust reports, they compress it and thereby lose information. It may be worthwhile to take a step back and see what should not be thrown away.
Many trust models (e.g. Jonker and Treur [1999]) model trust as a simple scalar $\in [0, 1]$ or $\in [-1, 1]$(where 1 denotes full trust). In his development of subjective logic, Jøsang [1999] argues that such a simple representation ignores one important dimension: On how much experience is the report built? How often has it been updated with new experience? In other words, how certain can you be about your opinion? Clearly, it is preferable to rely on a trust opinion which has been updated through experience 30 times than on one which has only been updated 3 times. A trust representation which incorporates this knowledge discounts the belief in a good outcome by the uncertainty that comes with sparseness of experience.
To this end, Jøsang (e.g. Jøsang [1999], Jøsang et al. [2006]) combines baysian modeling with belief theory. He updates trust by statistical updates to a PDF (Probability Density Function) in the light of new, binary (good or bad) experience. The more

important step is that he employs belief theory, in which the sum of all probabilities does not necessarily add up to one. The missing difference denotes the uncertainty, and it should become smaller by accumulating more experience.
The most basic notion is that instead of representing trust with a simple scalar between 0 and 1 (e.g. 0.7 would denote a belief that a positive experience is expected with the probability of 0.7), which divides the possibility space in two, a three-fold *belief space* is modeled. The third fold takes uncertainty into account and helps to distinguish on how much experience a trust report is actually based. See Figure 2.2 for an illustration.

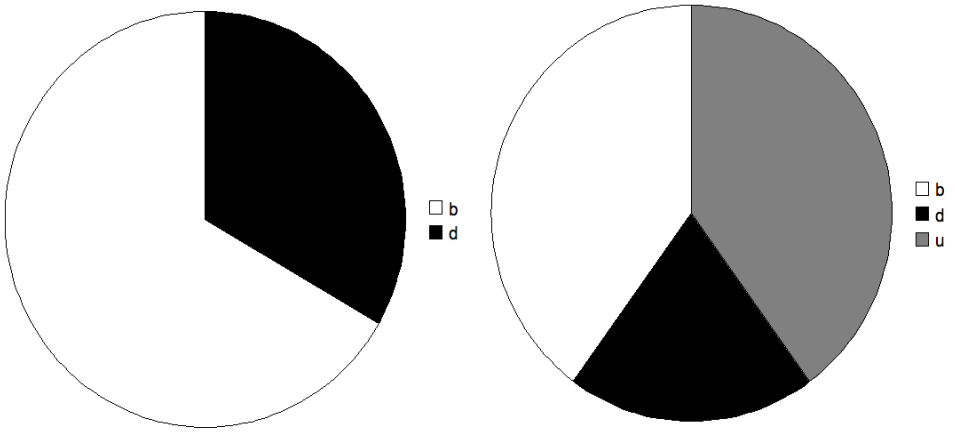


Figure 2.2.: Scalar trust vs certainty-based trust with uncertainty. The ratio of belief and disbelief is equal in both charts.

A more complex representation will of course also open new questions. For instance, how to translate actual experience, the *evidence space*, into this belief space? How should two of such trust reports be combined? What should the certainty be when an agent accumulated a lot of experience, but it is not very conclusive (similar amounts of positive and negative experiences)?

We will now quickly summarise the recent history of this research path and then go into details concerning the existing work in certainty-based trust. While Jøsangs work was influential, the researcher group around Singh (e.g. Yu et al. [2004], Wang and Singh [2006]) has developed this concept further and answered many practical questions. Therefore, the presented model will follow closely to Wangs doctoral thesis (Wang [2009]).

2.2.1. A brief history

As has been noted above, Jøsang [1999] establishes the certainty-based trust models we introduce in this chapter. In particular, he describes two trust representation spaces: an evidence space, which simply holds good and bad experience counts, and a belief space, which also incorporates uncertainty. He proposes how to translate between these two

spaces and introduced two operators with which trust reports could be combined in place or along referral paths: consensus and conjunction, respectively.
Wang and Singh [2007] then attempt to formally ground Jøsangs trust notion to make it handle more situations correctly. They adjust evidence transformation such that

1. that Laplace's *rule of succession* holds
2. the certainty decreases as conflict increases, provided the amount of evidence is unchanged

To this end, they also introduce a new bijection between evidence and trust space.
Later, Wang and Singh [2006] define two operators for their slightly different representation: concatenation (refining Jøsangs conjunction operator) and aggregation (refining Jøsangs consensus operator). We will go into details about operators in section 2.3.3.
Hang et al. [2008] put this framework to the test in a trust network simulation. They present a simplistic testbed with agents that change behaviour, leading to conflict in the data, in order to demonstrate how their model can handle such situations. We use their model as a starting point for our own in chapter 4. They also develop an update operator to update the trust in agents based on the accuracy of their referrals.
Finally, Hang et al. [2009] test their algorithms among several others on two social data sets. They add a selection operator in order to avoid double-counting of information and use the Weber-Fechner law to adjust subjectivity in test data sets while transforming them into evidence space

2.2.2. Belief Models

2.2.2.1. Evidence Space

To keep things simple, trust is modeled in a binary world of good or bad experiences. An agent counts his positive and negative experiences r and s, respectively. This forms the database of his experience $\langle r, s \rangle$.
Normally, an agent would increment r by 1 if his experience is positive, and s by 1 in the opposite case. Fuzzy representations of experience are possible: The experience database is allowed to contain real numbers. This makes flexible discounting mechanisms possible. We introduce here the discounting parameter $\beta \in [0, 1]$. When updating the evidence with new information r' and s', all previous information r and s can be discounted by β:

$$r = r' + (1 - \beta) * r$$

$$s = s' + (1 - \beta) * s$$

Intuitively, $\beta = 1$ forgets all old information and $\beta = 0$ forgets nothing [2].
The maximum probability of a good outcome α is denoted with $\frac{r}{r+s}$. If $r = s = 0$, α is set to 0.5. [3].

[2]The issue of discounting is of central importance for this research and will be revisited later.
[3]Often, Laplacian Smoothing is used in these cases where both sides of the fraction have default additions. Then, α would be $\frac{r+1}{r+s+2}$

2.2.2.2. Belief Space

JØSANG [1999] developed a so-called "belief space", where the agent now denotes belief (in a good experience) as $b \geq 0$, disbelief as $d \geq 0$ and the remaining uncertainty as $u \geq 0$. Adding up b, d and u yields 1 and the certainty then is $1 - u$ or $b + d$. To reach the belief space $\langle b, d, u \rangle$ from the evidence space $\langle r, s \rangle$, we first compute the certainty.

2.2.2.3. The Probability-Certainty Density Function

Probability of binary events can be modeled by a probability density function (PDF). Figure 2.3 shows a typical example (after collecting 1 negative and 8 positive experiences). When nothing is known, the curve will resemble a uniform distribution over all probabilities p. The probability which is most likely is $p = \alpha$, so the PDF reaches its maximum at this point. So in the normally distributed case, when nothing is known, α is 0.5. The more the data suggests a certain α as likely outcome, the sharper the curve will become.
This models an *a posteriori* trust distribution. When qualities are described by a distribution rather than a single number, this means that evaluations can be made on the basis of the perceived mean and variance of a reputation rather than an absolute number.

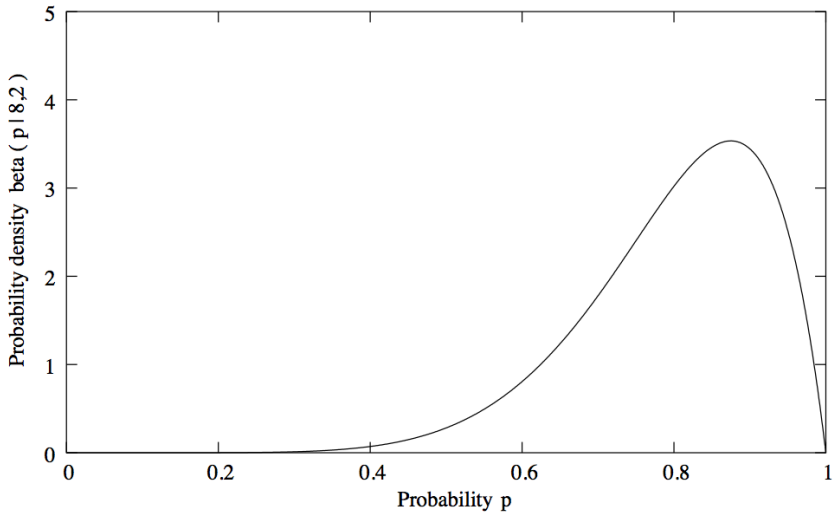


Figure 2.3.: A probability Density Function, from JØSANG ET AL. [2007]

Wang and Singh formulate a domain-specific probability-certainty density function (PCDF). The certainty (given r and s) then is

$$certainty(r, s) = \frac{1}{2} \int_0^1 |\frac{p^r(1-p)^s}{\int_0^1 p^r(1-p)^s dp} - 1| dp$$

Wang and Singh arrive at this formula in several steps, but the intuition behind it is that the certainty is higher for bigger deviations of the PCDFs integral from a normal distribution.
To this end, they compute the conditional binomial $f(p|\langle r, s \rangle)$ (CASELLA AND BERGER

[1990], p. 298) for any p in order to accumulate all mean absolute deviations from the mean value of the integral over any PDF (which is 1)[4]. Every increase in the PCDF for p is a reduction in another PCDF for p', so they remove double counting by multiplying by $\frac{1}{2}$.

2.2.2.4. Dealing with Conflict

Contrary to the belief representation defined by Jøsang, Wang and Singhs representation paid attention to conflict in the data. They denote conflict in the experience with $min(\alpha, 1-\alpha)$. Their intuition is that certainty usually increases when the amount of data increases. But we should also care for the amount of conflict in that data. If $\alpha = 0.5$, we can't be as certain as when $\alpha = 0.9$. The discussion of the probability-certainty density function above indicates that the deviations in a conflict-rich experience lead to lower certainty.
More formally, Wang and Singh point out two properties of their model: When the amount of experience increases with fixed α (e.g. from evidence $\langle 2, 8\rangle$ to $\langle 20, 80\rangle$), the certainty also increases. When the amount of data stays the same, but the conflict in the data increases (e.g. from evidence $\langle 2, 8\rangle$ to $\langle 5, 5\rangle$), the certainty decreases.

2.2.2.5. Transferring between Spaces

The way in which the belief space is computed from the evidence space is crucial, as becomes evident by the discussion above. It is also desirable to provide a bijection between both spaces.

From Evidence to Belief With the certainty calculated from the evidence, we can compute all parts of the belief space. The belief is $certainty * \alpha$ and the disbelief is $certainty * (1 - \alpha)$. Basically, we discount the belief and disbelief by the certainty. The uncertainty then simply is $1 - certainty$.

From Belief to Evidence Wang and Singh [2007] provide a projection from belief space to evidence space [5]. Basically, the amount of evidence $t = r + s$ is estimated from $\langle b, d, u\rangle$ until $certainty(r, s)$ is close enough to $b + d$. Since we know $\alpha = \frac{r}{r+s} = \frac{b}{b+d}$ [6], we can compute the evidence for each t as $\langle t * \alpha, t * (1 - \alpha)\rangle$.

[4] For practical purposes, they discretise over n intervals between 0 and 1 for the integrals over p, e.g. $n = 1000$

[5] Jøsang [1999] also provided one for his trust representation

[6] $\frac{b}{d}$ represents the same ratio as $\frac{r}{s}$ as they both represent $\frac{\alpha}{1-\alpha}$

2.3. Referral Networks

In this section, we describe the notion of referral networks. We list ingredients of such systems, explain how referral trees are built and explain the operators being used to aggregate, concatenate and update trust reports.

When agents make use of the trust reports provided by other agents, they form a referral network. A referral network is an information dissemination system consisting of autonomous and possibly self-interested agents. Trust reports are the protocol of choice among such agents and as we have seen in the previous section, there are many ways in which this protocol can be realised and what it actually talks about.
A referral is provided by agent B to agent A by giving A a report about his trust in C (see Figure 2.4). The path from A to B to C is called a referral path (and could it principle also extend over several other agents).

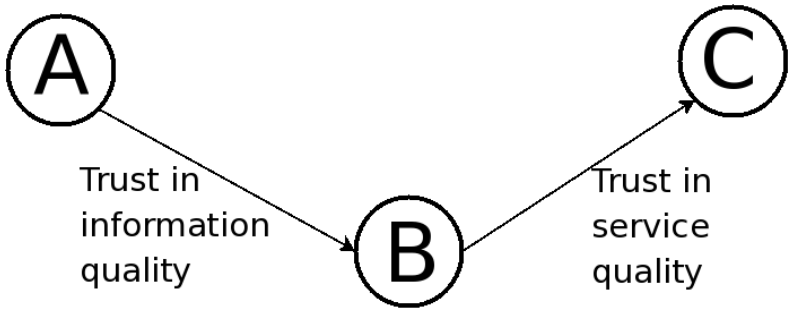


Figure 2.4.: A simple Referral Network

The main task of a referral network is to disseminate information encountered in direct experience, which thus becomes indirect knowledge to its recipients. Relating to our discussion in section 2.1, trust (direct experience) becomes reputation (indirect knowledge). However, this reputation is not of a global nature, but a local view, dependent on the information sources each agent has. An effect of this creation of reputation in complex social systems is often the emergence of cooperation through indirect reprocity. Nowak and Sigmund [2005] put it like this:

"Presumably, I will not get my back scratched if it becomes known that I never scratch anybody elses. Indirect reciprocity, in this view, is based on reputation."

Referral networks have been studied widely (e.g. Singh and Yolum [2003], Ding et al. [2005]). In this section, we will describe briefly the ingredients usually required for a referral network and in particular describe the operators with which agents combine trust reports along referral paths. Here, we will in particular refer to work done by Jøsang et al. [2007] and Wang [2009].

2.3.1. Ingredients

Adaptability The key advantage of a referral network is its adaptability to reflect changing circumstances in the environment (where other agents are also counted within the environment). By learning from other agents what they experienced, each agent increases his sensual radius. Furthermore, each agent can adapt on its own. If an agent knows more than one peer, it can decide which agent to contact and thus gains flexibility.

Autonomy From the previous point follows that agents should be able to act autonomously in their own regard. They are able to refuse service or provide bad service intentionally. Other agents should not rely on cooperation. This models undetermined and dynamic environments and enables agents to react quickly to perceived changes.

Weighting A referral can in principle be provided without any personal opinion in the trustworthiness about the referee. Then, a referral is nothing but a connection. However, from the principle of autonomy follows that trust is necessary to rate the suitability of a referral. If each referral is provided with a trust report attached, we can regard the referral network as a weighted graph.

Trust semantics Note that different trust semantics are involved in the above example. While the trust which B has in C talks about the service quality C can provide, the trust A has in B talks about the quality of information that B provides. Yu and Singh [2002] call these two notions *expertise* (quality of service) and *sociability* (quality of information). When combining trust reports along a path, caution should be taken to this distinction.

Agent Roles Agents can take on different roles in a referral network. When an agent asks the network for an opinion, we call him a *client*. The client in the introductory example would be agent A. The agent being asked about is called the *service provider*, which in the example would be C. If an agent responds to a request with the opinion based on his own experience, he acts as a *witness*. In the example, B is a witness. When an agent doesn't have an opinion about the service provider (or would rather not give it), he can merely refer to another agent. In that case he acts as a *referrer*. Note that the example has no agent in this role. Figure 2.5 shows an extended example, in which agent B acts as a referrer to D, who in turn acts as a witness now.

While it is natural to assume that each agent can play all roles in a given system, it can make sense for a research setting to allow agents to play only one role (which is what Hang et al. [2008] and this research do). This way, we avoid any interdependencies from the different roles (see also the discussion on complexity and interdependencies below).

Protocol In essence, any kind of agent can be part of a referral network. What matters is the protocol the agents use to refer to each other. Thus, the protocol is defined by the representation used for trust. A certainty-based trust report like we use here is not

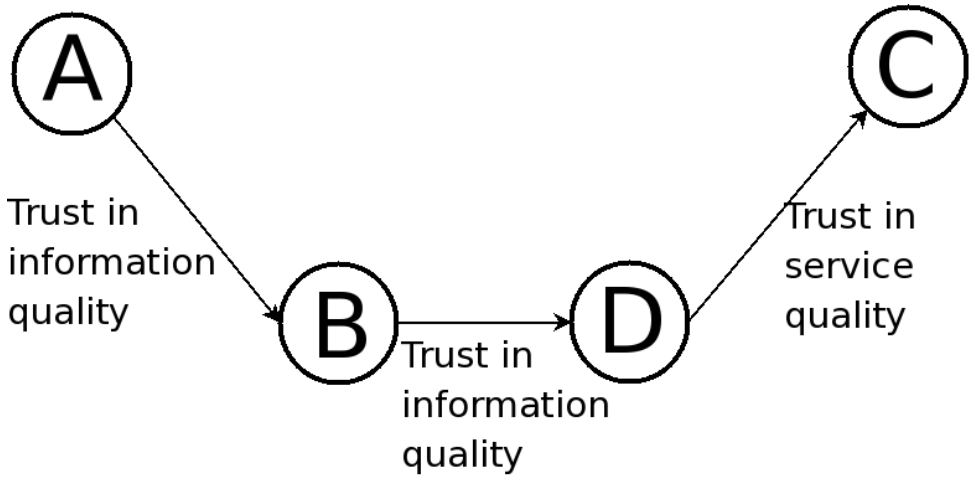


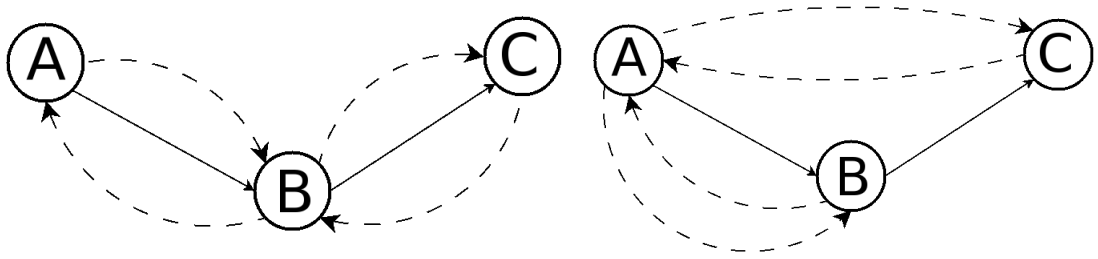
Figure 2.5.: A simple Referral Network with four agents

compatible with a simple scalar trust report. Interestingly, this incompatibility is one of the main reasons why Hang et al have not tried to design an agent for the ART trust competition (Fullam et al. [2005]).

Purpose The general design purpose of a referral network is to enable an agent to receive an estimate about some service from the network. The service in question could be anything from providing a valid response to be truthful about its identity. It could be another agent in the network or an external entity. The specific incentive for an agent to use trust reports from the network (for instance, to make less errors or receive more payoff) has no impact on this general idea.
A referral network design should provide good estimates and in particular be resilient against fluctuations in service quality and malicious behaviour of other referrers.

Message Routing When we say that an agent speaks to "the" network, we could more fittingly speak of "his" network. Since in most networks not every agent is connected to every other one, each agent has a unique standpoint in the networks structure and his local view is distinct to that of every other agent. The structure of networks thus has a significant effect on the quality of referrals. A very popular network structure is the Small-World network (Watts and Strogatz [1998]), which resembles the social structure found in most human and many biological systems.
Another design aspect concerning structure is how agents should refer along a path. One approach (e.g. Höning et al. [2008]) is to let agents refer *recursively*. Look again at Figure 2.5. Here, agent A asks B about C, which in turn asks D about C. The trust report is then propagated back to the beginning of the path (and maybe modified each time), so D gives his report to B, which gives a report back to A. In contrast, the *iterative* approach lets agent A be involved in every step, so A asks B, who gives him a trust report referring to D. Then A asks D, who gives him his trust report about C. One advantage of the iterative approach is that agent A has complete control over each step. He can

decide if the path is worth pursuing and he can update its own trust in the referrers after he evaluates how fitting their reports were. Also, referrers will operate in a much simpler and well-defined context. They only know the source and the target of the path which makes modeling them much easier.
The figure below illustrates the message routing in both approaches for the simple network from Figure 2.4.



(a) recursive approach (b) iterative approach

Figure 2.6.: Message routing in referral network designs

Node Routing Each referrer has the choice to route his referral to any of his acquaintances. He might choose the one he trusts the most. He might prefer to refer to witnesses over referrers, even if he trusts the latter more. He might also choose to provide more than one referral.
As we discuss these issues, we move into the strategy space - especially if referrers have their own incentives in providing good or bad referrals [7].

Complexity Work on referral networks (and this research is no exception) has to face the complexity that comes with the interdependence of its components. When subjective information (trust) turns into objective information (reputation), it becomes harder to tell where information came from or even when it was created. Some think that systems should be limited in their ability to build referral chains. For instance, JØSANG ET AL. [2007] write:

"In order to avoid dependence and loops it is required that referrals be based on first hand experience only, and not on other referrals."

However, even they continue with noting working examples, for instance Googles PageRank algorithm (PAGE ET AL. [1998]), which normalises over referrals of each agent [8]. In essence, every model will use some restrictions to tackle the complexity, in order for the researchers to understand it. Which ones to restrict may be open for debate. It might be interesting to study how humans do it. How far can reputation travel among

[7] This topic is therefore not addressed in this research. All referrers will simply give one referral to a random neighbour, preferring witnesses over other referrers.
[8] It could also be argued that introducing uncertainty is a way of normalising, but this is done internally by each agent.

humans? How long does it survive? Of course, the complexity of human social systems makes it hard to give definitive answers to these questions, so we might end up on square one.

2.3.2. Building Referral Trees

When a client asks referrers for an opinion about a service provider, he starts building a so-called *referral tree*. Each referrer may refer to one or more witnesses or even to other referrers, which brings about a branched structure of referrals. Each referral is annotated with a trust value - the meaning of which depends on the role of the referred agent: it might mean how trustworthy the information from the next agent on that branch is or how likely he is to provide good service.
Via iterative routing, the client asks each current leaf on the tree to provide new referrals [9] and adds them as new leafs on that path, unless the referral comes from a witness and thus refers to the service provider.
Once all branches have service providers as leaves, all trust values get combined as the goal of building such a referral tree is usually to end up with one trust value which represents the opinion of the clients network. To this end, trust values have to be concatenated along the paths and aggregated at each branch. We need to define operators for this which the next section has more details on.

2.3.3. Operators

In any referral network, combining trust reports from multiple sources is important, but non-trivial. An agent needs to do this when he builds a referral tree or when he updates trust values based on feedback. To design these operators is one of the main tasks when planning a referral network. We will here formalise the operators used in a certainty-based trust representation (see section 2.2).
The two most basic operators, aggregation and concatenation, had already been proposed by Jøsang et al. [2007]. Wang and Singh [2007] only adjusted concatenation and also, in later papers, proposed two new operators, update (Hang et al. [2008]) and selection (Hang et al. [2009]). Below, the operators as developed in Wangs, Hangs and Singhs work are explained.

2.3.3.1. Concatenation

Along the same path, trust is propagated using the concatenation operator. Let $x.belief$ denote the belief computed on trust x. Trust a $= \langle r_a, s_a \rangle$ concatenates with trust b $= \langle r_b, s_b \rangle$ to

$$\langle a.belief * r_b, a.belief * s_b \rangle$$

Intuitively, the experience on trust b gets discounted by the belief trust a puts in trust b. So the witness information from the end of a path travels to the beginning, being

[9] In this work, agents will only give one referral.

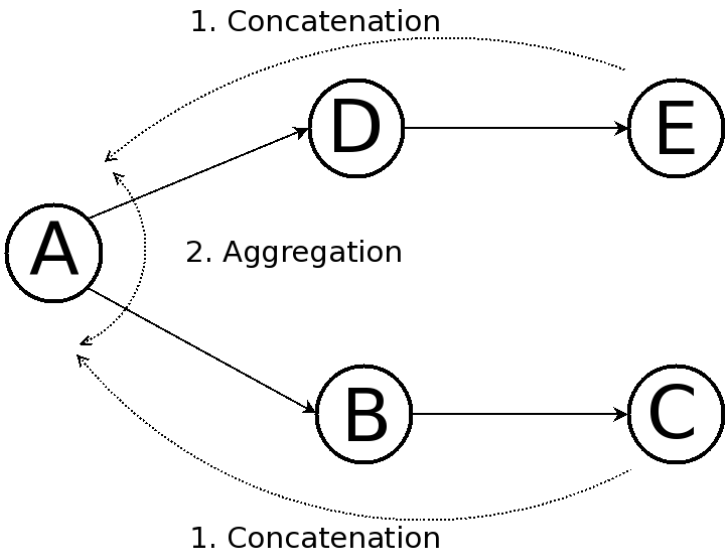


Figure 2.7.: Ilustration of concatenation along paths and aggregation of the results

discounted along the way. The concatenation operator is proven to be associative and commutative.
It is important to notice that referrers have a limited role by use of this concatenation operator. Every referral path only provides the original trust report of the witness at its end, discounted by the referrers beliefs. The referrers can not change the probability α given by the witness, because α is based on the ratio of r to s and with every concatenation r and s both get discounted by the same amount: the belief of the referrer[10].
The function of a referrer, then, is twofold: First, he has to choose which witness or other referrer to refer to. Second, he should estimate if the reports he gets are likely to be truthful or not and place his referral trust accordingly [11]. In an extreme case, where only the false reports of defecting witnesses are available, the referrers can do nothing but soften the impact of this situation by adjusting his referral trust.

2.3.3.2. Aggregation

Aggregation happens when different paths between the same target and source get merged. Trust a $= \langle r_a, s_a \rangle$ aggregates with trust b $= \langle r_b, s_b \rangle$ to

$$\langle r_a + r_b, s_a + s_b \rangle$$

Intuitively, this simply adds up both experiences in evidence space. The aggregation operator is proven to be associative and commutative.

[10] However, if one of r or s is zero and the other is not, then only one of the two actually gets discounted. Note that this does not affect the ratio, since if one of r or s is zero, the ratio is undefined anyway.
[11] We will revisit this two regards of a referrers ability when we reconsider the update operator in chapter 4.

2.3.3.3. Updating

With the update operator, an agent updates his trust in a referrer based on the usefulness of the information his referral provided. For instance, the client computes the ratio between the trust he now has in the service provider (after the service provision) and what the referrals given by the referrer provided as opinion. By definition, this takes place ex post, after the client got the service. See Figure 2.8 for illustration.
Let td be the clients direct experience and tr be the concatenated trust from the path which the referrer provided. Also, given any trust x, $x.\alpha$, $x.r$ and $x.s$ denote the α, r and s of trust x, respectively. The accuracy q is computed as

$$\frac{tr.\alpha^{td.r} * (1 - tr.\alpha)^{td.s}}{td.\alpha^{td.r} * (1 - td.\alpha)^{td.s}}$$

The formalisation of this accuracy ratio q is developed from the probability-certainty density function, which was explained above. Note that q is computed in evidence space only, so the certainty is not important for q.
The client can now update his trust in the referrer $tref$:

$$tref.r = tr.certainty * q + (1 - \beta) * tref.r$$

$$tref.s = tr.certainty * (1 - q) + (1 - \beta) * tref.s$$

Note that we use the discounting parameter β, since the update operator is time-oriented by definition. Wang [2009] also suggests some other variants of this operator to better accomodate against malicious referrers, very steep PCDFs and uncertainty.

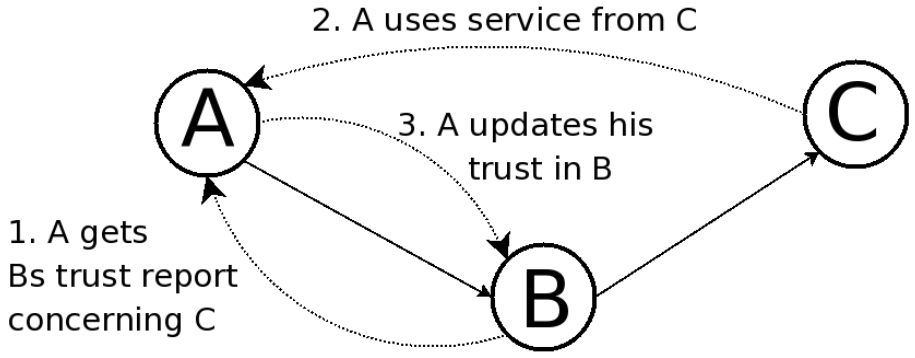


Figure 2.8.: Illustration of the update process

2.3.3.4. Selection

In passing, we discuss an operator from more recent work in Hang et al. [2009]. They introduce a selection operator to tackle double counting of information. To a client who asked two referrers, it may seem that he aggregates information from two paths while really the two referrers may in turn have returned information from the same witness.

This would lead to double-counting of the information the witness gave.
Performing a selection among paths means that the client compares two paths and discards the one that is less reliable (in terms of concatenated beliefs). The selection operator is proven to be associative. While it worked well, an empirical study on real datasets could not prove that their selection operator performs significantly better that a simple model.

2.4. The value of information over time: A trade-off

In this section, we consider how the concepts of certainty and recency develop with the lifetime of a system. We look at techniques used to realise discounting and dynamics that occur in systems in which agents try to maximise both concepts.

We discussed how a trust system could model uncertainty in section 2.2. If we imagine agents that accumulate information over time, than certainty would increase steadily over time. However, while time passes, things in the environment may change. This thought introduces the concept of recency. An information can be recent or old. If we assume that changes happen for a reason and that changes develop continually, then clearly more recent information is more valuable. Both certainty and recency are goals when dealing with information, but they are mot complementary - it is a complex task for a decentralised system to find the right trade-off for each situation.

2.4.1. Recency: Exploring dynamic environments

The most basic approach to incorporate recency in ones memory is that old experience should be accounted for less, since things might change over time. Discounting mostly happens with a discounting factor like the one we introduced in section 2.2. For instance, here is the formula used by Jonker and Treur [1999] for a simple scalar trust representation:

$$g_d(tv, ev) = d * tv + (1 - d) * ev$$

where g_d is the update function which updates the trust value $tv \in [-1, 1]$ in light of a new experience $ev \in [-1, 1]$, using a discount factor $d \in [0, 1]$. The existing tv is also discounted to normalise the result of g_d in the range $[-1, 1]$. This is similar to our model, only that we update positive and negative experience separately and our discounting factor is called β.
The choice of the discounting factor is crucial and should depend on the situation. Hang et al. [2008] experimented with varying discounting factors in different situations (as we will further explain in chapter 4), but we have not come across any research in which agents choose their discounting factor themselves.

The strive for recency has also been encountered in Psychology. For instance, it is known as the "recency effect" (e.g. Baddeley and Hitch [1993]) that the most recent memories have an even stronger recall than linear discounting would predict.
Throughout the computer science literature, authors use different names. Instead of a "discounting factor", they might speak of a "forgetting factor" (e.g. Jøsang and Ismail [2002]), "aging factor" (e.g. Buchegger and Le Boudec [2003]) or "fading factor" (e.g. Buchegger and Le Boudec [2004]).

There are other techniques than to discount all history. For instance, in Keung and Griffiths [2008] and Huberman and Wu [2003] , agents update their trust in others while keeping a "history window" of previous interactions, the size of which is a controlled parameter.
Furthermore, discounting can happen not within each agent but at a more central location, for instance by the agent who receives the trust reports, like in Jøsang et al. [2006]. The latter lets all data be stored forever, since discounting only happens when needed, but has the disadvantage that the age for each rating needs to be known, opening the door for deception.

2.4.2. Certainty: Exploiting stable environments

Agents generally strive for more information in order to learn about their environment. When they are more certain in their actions, they can take higher risks and maximise their profits. The trust model described in section 2.2 builds up certainty over time, as agents interact.
However, when stored experience gets forgotten, the certainty also decreases. As agents in a trust system try to maximize both of these values, a trade-off situation occurs: How much should old experience be discounted in order to to benefit from both certainty and recency?

Several researchers have noticed this issue (while none that we know of have focused their analysis on it). For instance, Keung and Griffiths [2008] note the recency/certainty trade-off when agents produce less failures with smaller history windows, but become more vulnerable to malicious behaviour which exploits small history windows. In Huberman and Wu [2003] a simple reputation system is modeled, in which two firms and many customers were present. Customers discount their memories of direct experience and firms choose how to invest in their next intended quality level, based on their current reputation. They show that discounting is a necessary condition for equilibrium points to be reached but that uncertainty makes it unlikely that the system reaches any equilibrium point.
Huberman and Wu [2003] also note how time delays in the turnover from observation to action destabilises their system, which can be understood as an argument for a recency effect in customers.

To accumulate certainty is generally rewarding, unless the environment becomes too uncertain for accumulated memory to yield any valuable predictions. In that case, a focus on certainty can even be harmful as the agent(s) will be too slow to react to a disruption. Harrington Jr [1998] shed light on the welfare of a system in unstable conditions. They model a social system as a tournament system in which agents stay as long as they are more successful than a random opponent up to a maximal number of rounds (this is intended as a metaphor for hierarchies with promotions). Each round, one of two strategies can be used. They employ two types of agents: rigid and flexible. Rigid agents always apply strategy A or B throughout their lifetime, while flexible agents always decide between one of them, after evaluating the circumstances. Agents are rewarded for

playing the strategy most fitting to the situation, but it is also assumed that agents who played one strategy constantly become better at it and yield higher outcomes (which is an analogy for our concept of certainty).
Not surprisingly then, the environment decides which agent type is favoured. Stable environments favour the experience of rigid agents. The more disruptions in the circumstances trigger it to change its preference for strategy A or B, the better flexible agents fare. When a punctuated-equilibrium pattern of stable times and sudden changes is modeled, the flexible agents succeed only in times of disruption. As soon as stability is back, rigid agents rule again (though the individual agents of course might have been replaced). Harrington concludes by observing that the long-term success of any organisation could be predicted by the ratio of flexible agents the system manages to keep around in spite of stable times. If flexible agents become extinct, a disruption is much more harmful for the overall system performance.
In this case, we do not need to speak of recency versus certainty. To some degree, a system should manage to have both.

3. Objectives

In this chapter, we describe the objectives of this work. We state how certainty and recency are important parameters for the analysis of next-generation multiagent systems. We lay out our research questions concerning the effects of individual discounting strategies on system performance.

This work models a referral system in which agents handle certainty-based trust and takes a closer look at the dynamics of individual discounting strategies in disruptive environments. Here, we want to lay out the reasons why this is beneficial for the understanding of information systems. We describe our work in terms of testbed development and exploring the effect of parameter settings. Furthermore, we will state our research objectives in terms of research questions and hypotheses. We are interested in the relation between uncertainty in the environment and discounting strategies as well as self-organising patterns of discounting.

3.1. Applications

The previous section laid out how any information system should over time strive to build up certainty, but at the same time still be able to react dynamically. In a system consisting of autonomous agents, each agent can decide how he handles this trade-off, i.e. how much information he should discount in order to maximise his utility.
Interestingly, this problem has a local and a global perspective, since these local decisions will in some way influence the system utility as a whole. In our view, this is a design problem which has not been addressed so far, at least not in the context of referral systems.

In which research contexts is the addressed design problem of interest? According to Jøsang et al. [2007], the purposes of research in trust and reputation systems should be to:

a. "Find adequate online substitutes for the traditional cues to trust and reputation that we are used to in the physical world, and identify new information elements (...) which are suitable for deriving measures of trust and reputation."

b. "Take advantage of IT and the Internet to create efficient systems for collecting that information, and for deriving measures of trust and reputation, in order to support

decision making and to improve the quality of online markets."

In modern multi-agent systems, agents are increasingly designed to be autonomous and highly dependent on one another at the same time. This holds for still rather abstract ideas like trading agent systems (Arunachalam and Sadeh [2005]), which are still highly researched. It is, on the other hand, already a valid concern for systems who are closer to the light of day like Peer-To-Peer systems, in which trust is earned through providing upload bandwidth in local interactions, wireless sensor networks (Guangjie et al. [2007]), in which sensors communicate decentrally and every sensor should try to route information most efficiently, or web services (Maximilien and Singh [2004]), who use each other to fulfil service requests.
Clearly, agents in all of those systems will use some sort of trust to enable their decision-making and will have to deal with the fact that conditions (i.e. their counterparts) are subject to change. These are really the only two preconditions for this design problem to arise.
As agents are autonomous, the tools which a system designer can use are limited, but the protocol is an important factor. We hope to have made the case for a certainty-based trust protocol in this regard, thus fulfilling the first of Jøsangs purposes (which is a prerequisite for the second). Furthermore, we consolidate the development of a testbed in which these systems can be simulated and analysed, fulfilling the second purpose. We will now describe both of these objectives, testbed development and research, in more detail.

3.2. Testbed Development

The short paper by Hang et al. [2008] made a first approach to establishing a testbed for this kind of problem. We take their work as a starting point and offer the following contributions (as laid out in chapter 4):

1. The recreation of their experiments.
2. A suggested refinement to the update operator.
3. Definitions of in- and output behaviour: service providers with controllable severity of disruptions, a client risk behaviour according to referred certainty, and feedback into the system by the client.
4. Autonomous referrers who manage trust themselves.
5. Local discounting strategies for the client, referrers and witnesses.

Furthermore, we will publish the code we developed for these experiments in order to be used for further experimentation (see the appendix).

3.2.1. Research Questions

We conduct a series of experiments in order to answer the following research questions:

Which parameter settings are important? We want to adapt the parameters which are influential for performance[1] and point to conflicts between settings.

Can the performance stabilise in highly disruptive scenarios? Service disruptions mean that the certainty decreases quickly and thus agents will reduce their stake (and rightly so). However, after a disruption, it would be good for performance if certainty levels could rise quickly. Viewed on a longer time scale, If the system suffers high disruption levels, it would be desirable if performance could be predictable (if deviations could be held at bay).

What is the effect of differing discounting strategies? The research by Harrington Jr [1998] indicated that a system fares best when it keeps both exploiting and exploring agents around. On the other hand, one could argue that alignment effects favour systems in which agents use the same discounting strategy.

3.2.2. Hypotheses

We state a list of hypotheses, to which we will refer when we discuss experiment results.

1. Parameter exploration:
   a) Our referral systems can accept high percentages of defective agents without a negative performance.
   b) The structure parameters of the network (neighbourhood size, ratio of witnesses to referrers) are significant to the performance as they influence the average path length.
2. Environmental effects on discounting:
   a) The environment influences what discounting strategies work best.
   b) Low discounting factors are hurtful in highly disruptive scenarios.
3. Performance benefits if agents are free to adjust their discounting factor.
4. Alignment of discounting factors:
   a) The alignment of discounting factors among agents is crucial for performance.
   b) It is good for system performance if agents employ different discounting factors.

[1] With performance, we mean the ability of the system to provide correct and certain assessments.

4. Model

In this section, we describe the details of our referral network model and implementation. We first define basic characteristics. Then we explain the certainty-based network model of Hang et al. *[2008] and his experiments (which we recreated). Finally, we formalise our own contributions and extensions to the model which are of importance to our own experiments.*

The experiment setup in this work is based on experiments on the certainty-based trust representation in a referral network devised in Hang et al. [2008]. We will introduce this work here first and then elaborate on the differences and contributions this work makes to the model. This is part of the consolidation phase of this work.
Though some details about the implementation were extracted from the code that was kindly provided by Hang et al, all code needed for this work has been written from the ground up[1]. The graphs shown here are produced by our own recreation of Hangs environment (but correlate to the ones found there). We also note that all experiments we mention here and in the experiment chapter are run with the same codebase.

4.1. Characteristics

We will begin by describing the model we use in basic terms, thereby following Schut [2009]. We explain the basic workflow of a simulation run, the diversity of involved agents and the control loop of an experiment. We then mention which parts of our model are non-deterministic and adaptive.

4.1.1. Workflow

We model a referral system. A service provider provides a service, the quality of which can change between being good or bad. A client asks referrer agents he knows for information about the service quality. On each of those referral paths, he gets referred until he finds a witness. The trust value of the witnesses gets aggregated along the referral paths and modified by the referrers opinions about their referee (the agent they referred to).
Thus, information travels from the service provider as first-hand experience to witnesses. It then becomes reputation when the client incorporates the referees opinion. See figure 4.1 for a broad overview over information flow. Note that also the client gets direct experience. This is used to evaluate the reputation information he receives.

[1] See the appendix for online access to the code repository.

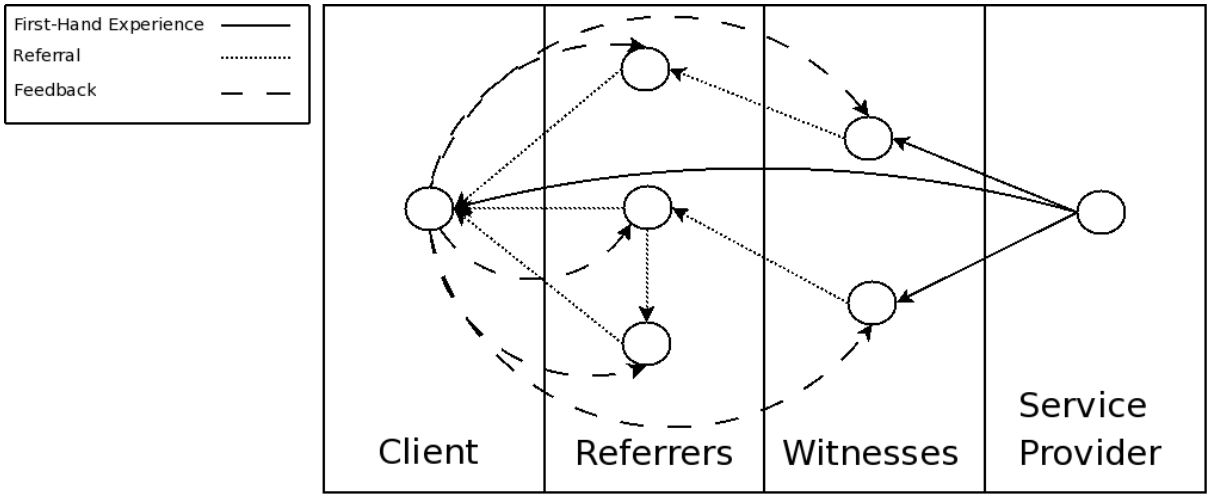


Figure 4.1.: Information flow in the referral system

4.1.2. Diversity

As has been mentioned, there are four agent types: Client, Service Provider, Referrer and Witness - all of which were introduced in section 2.3. Agent roles in the referral network are strictly distinct (we also discussed this design option in section 2.3). Table 4.1 notes their internal models, action and observation sets. When an item is noted in brackets, it is a feature we introduce in later experiments. We will explain the behaviour of each agent in more detail in the next section, where we will also introduce subtypes (e.g. bad referrers and witnesses, damping providers).

Type	**Internal Model**	**Observations**	**Actions**
Client	direct experience [cont.], trust in referrers and witnesses [cont.], β [cont.]	service, referrals	referral requests, (give feedback), (update β)
S.-Provider	service plan [discr.]	service requests	change service quality, provide service
Referrer	honesty of neighbours [discr.], (trust in referrers and witnesses [cont.]), (β [cont.]), (trust in own history [cont.])	referral requests, (feedback)	provide referrals, (update trusts), (update β)
Witness	Direct experience [cont.], β [cont.], (trust in own history [cont.])	service	provide direct experience, (update β)

Table 4.1.: Agent types and their internal models (and if the representation is discrete or continuous), actions and observation sets. β denotes the discounting factor.

4.1.3. Control Loop

Each experiment proceeds as described by algorithm 4.1. Again, brackets denote items that get introduced later on.

Algorithm 4.1 Control loop for simulations

```
initiate agents according to experiment setup
connect agents
service provider makes service plan
epoch ← 1
while epoch < runtime do
    witnesses experience service and update their direct experience
    client builds referral tree
    client experiences service
    client evaluates accuracy of referrals
    client updates his trust in referrers and his direct experience
    (client gives feedback to all agents in referral tree)
    (client, referrers and witnesses update their β)
    (referrers update their trusts in neighbours)
    epoch ← epoch + 1
end while
```

4.1.4. Adaptivity

It is important to note that the model by Hang et al. [2008] already is partly adaptive: The client updates his trust in referrers and both the client and witnesses update their own direct experience. However, referrers are modeled very simple. We will later make them more adaptive by having them keep and update trusts about their neighbours. In addition, we will let the client, referrers and witnesses update their discounting factor β.

4.1.5. Non-Determinism

Uncertainty is not only modeled by changes in service quality, but also by several randomisations. In addition, randomisation is a good mechanism to explore the solution space. We list them here to clarify the system behaviour:

1. Connectivity among agents: It is specified in the experiment setup how many agents of which type each agent knows, but it is decided randomly which ones.

2. Routing: A client will ask all the referrers he knows for a referral, but then all referrers will only pick one random neighbour to refer to. This mechanism is used due to simplicity and its ability to explore (and hence adapt) all possible referral paths.

3. Service quality in disruptive provider: In our own experiments, we model a service provider whose service plan is probabilistically determined by random disruptions, which shapes the environment for the system..

4. Deviations in simple trust reports: The simple, static referrer behaviour is to attach a good or bad trust report to the referral. In order to introduce some randomness, slight fluctuations are added to the experience of these trust reports.

5. Distribution of β: In the last experiments, we assign values for β to agents. Some runs assign values randomly to all agents.

4.2. Model of Hang et al (2008)

Hang et al. [2008] built a simulation testbed to compare the certainty-based trust representation and their operators against other operator models like that of Josang. The following diagram illustrates the layers and the communication structure:

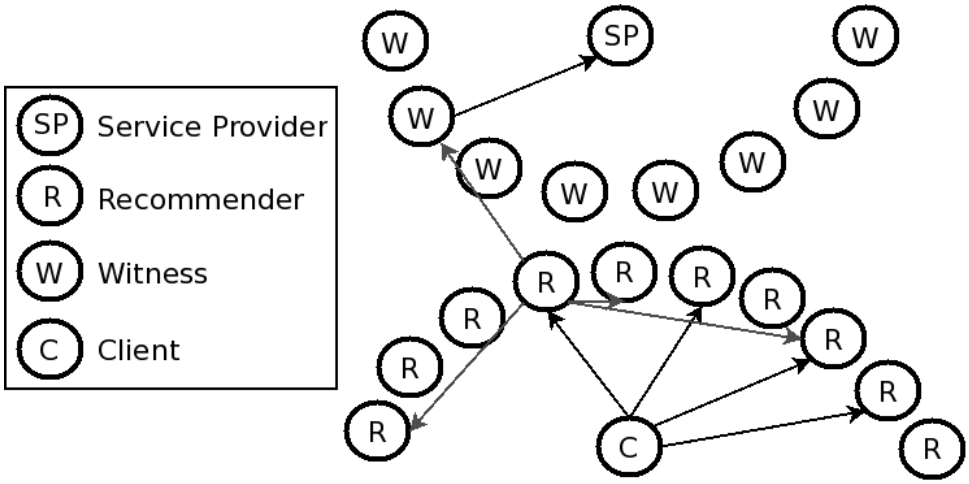


Figure 4.2.: Network structure in Hang et al. [2008]. Connections of one client, one referrer and one witness are depicted.

4.2.1. Agents

The Client There is only one client, who is connected to four random referrers in each simulation run. He keeps track of his direct experience with the service provider by a trust value in which he increments r or s for any positive or negative interaction, respectively. This trust gets discounted over time by the discounting parameter β. Also, he has a trust value for each referrer, which he updates with the update operator discussed in section 2.3.3 when he evaluates the quality of their referrals.

The Referrers There are ten referrers. All referrers know two other referrers and some know one or more witnesses. When asked for a referral, they will provide a referral to a random witness if they know one or a random referrer they know otherwise. This randomness makes sure that all agents take part in the information flow (knowing a witness gives a referrer a prominent position, but this is newly assigned for each run).
The trust opinion they attach to the referral is simple: If the agent they refer to is honest and they themselves are honest or if they both are dishonest[2], the referrer will have a good opinion ($\langle r, s\rangle = \langle 19,^{+}_{-} 1\rangle$ - $^{+}_{-}1$ means that there is some random variation). Otherwise, they will have a bad opinion (with r and s reversed). Referrers can be good (always honest), bad (always dishonest) or damping (honest, change to dishonest after 10 rounds).

The Witnesses There are eight witnesses (by having less witnesses than referrers, some referral paths will involve more than one referrer). Each witness is known by exactly one (randomly chosen) referrer. All witnesses know the service provider and keep a trust opinion about him. They interact with the service provider once each round and update their trust in him accordingly, just like the client. Witnesses can be good (always honestly returning their actual trust in the service provider when asked by a referrer) or bad (return the opposite trust $\langle s, r\rangle$).

The Service Provider For simplicity, there is also only one service provider and the service provided is either good or bad. This agent is equipped with a probability p to provide good service. A service provider can be good ($p = 1$), damping ($p = 0$ for the first half of the simulation, $p = 1$ for the rest) or capricious (p alternates between 0 and 1 every 2 rounds).

Note how referrers are rather simple here, only relaying information while acting on behalf of global information (the honesty of the referee). In contrast, the client and the witnesses show adaptive behaviour: The client updates his trust in referrers and both the client and the witnesses update their trust in the service provider.

4.2.2. Experiments 1-3

A simulation run takes 20 rounds. In each round, the client builds a referral tree (see section 2.3.2) using all referrers he knows and updates his direct experience with the service provider like explained above. He then updates his trust in the referrers with the update operator, using his direct experience and the referral tree. We do not know how many runs have been conducted for each experiment.
Note that we will, as a convention, draw a graph black if it denotes a factor completely controlled for by the experiment. In these experiments, this holds for the service quality.

The first objective is to evaluate if the opinion referred to the client by the network matches his personal direct experience. In particular, Hang et al were interested in situ-

[2]Note that this knowledge is made available to the referrer by default and not learned.

ations of conflicting information. These situations come about when the service quality changes or recommenders turn malicious. Their result is that with the operators defined by the authors (and discussed here in section 2.3.3) the trust referred to the client by the network models the actual experience better than with other operators. In particular, they modelled three scenarios:

1. In experiment 1, the service provider is capricious and half of the referrers and half of the witnesses are bad. In these uncertain settings, the referred trust manages to stay close to the direct experience of the client (see Figure 4.3). The discount factor β is 0.0, because its effects (the updating of trust in referrers) are not of interest in this experiment.

2. In experiment 2, the service provider is honest and half of the referrers are damping. All witnesses are honest. In three subsettings, three different values for the discounting parameter β were employed (0.0, 0.4 and 0.8). Figure 4.4 illustrates how higher values of β make the system more responsive, but uncertain. The authors conclude that 0.4 is an acceptable compromise.

3. In Experiment 3, the service provider is damping and the referrers and witnesses are honest. The clients discount factor β is set to 0.4. This is a showcase scenario where it can be seen that a clients trust in an honest referrer can stabilise, even in times of rapid change in quality. When the service provider changes its quality completely, then the client adjusts his trust in the honest referrer quickly after a brief period of dibelief and uncertainty. See Figure 4.3.

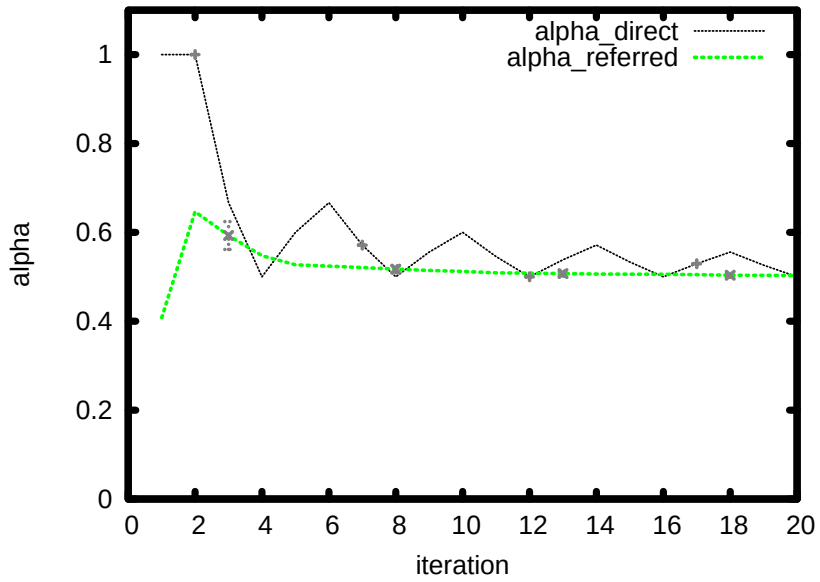
The general conclusion of Hang and colleagues is that the operators they defined enable a referral network to model trust accurately. The belief representation (in combination with the concatenation and aggregation operators) is capable of also modeling conflicting and disruptive situations well. Malicious referrers and witnesses will be less influential due to the update operator.

4.3. Outputs and Inputs

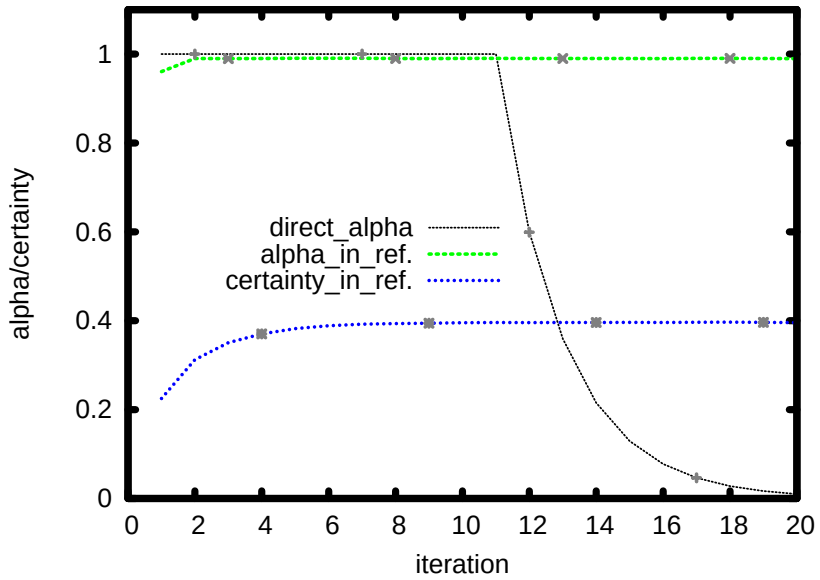
In this section, we clarify the notions of input and output to such a referral system and propose a model to simulate them. We view the service quality as an external input to the system and the clients utility as an output. Furthermore, the client might provide evaluations and feedback as input into the system. Figure 4.5 shows a rough system overview, modeling the referral system as the system under consideration, with input and output connected to the client and the service provider.

4.3.1. Input: Disruptive service quality

In Hangs experiments, the probability of a good service changes between 0 and 1, in alternating (experiment 1) or damping (experiments 2 and 3) manner. These changes

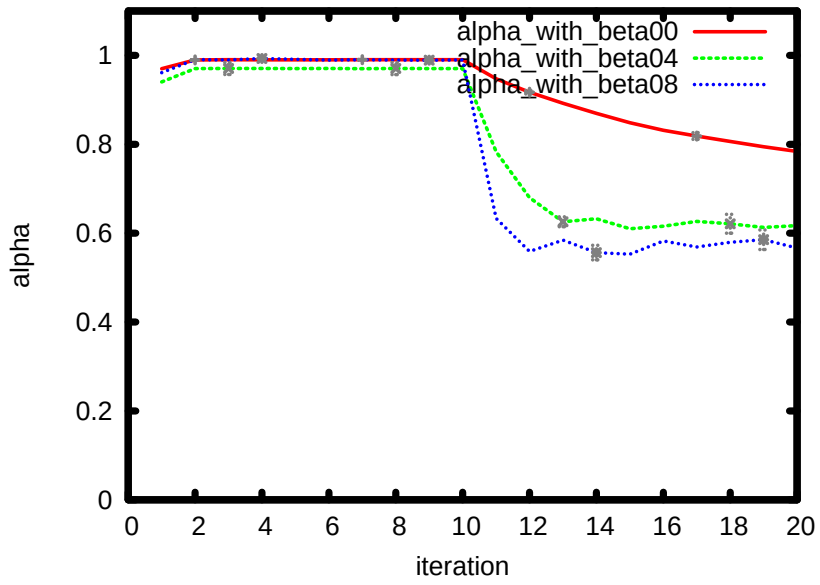


(a) Experiment 1

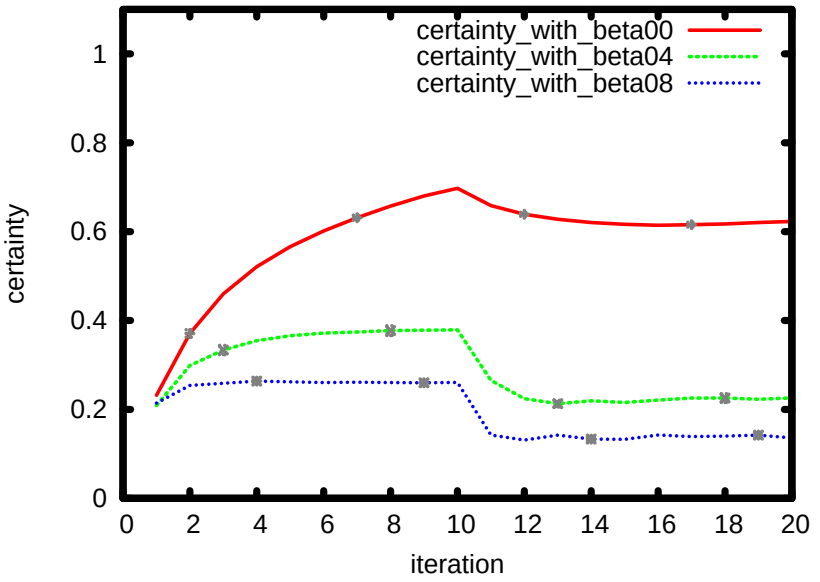


(b) Experiment 3

Figure 4.3.: Results of Experiments 1 (a) and 3 (b) in Hang (2008)



(a) α of the clients trust in a damping referrer for different values of β



(b) Certainty of the clients trust in a damping referrer for different values of β

Figure 4.4.: Results of Experiments 2 in Hang (2008)

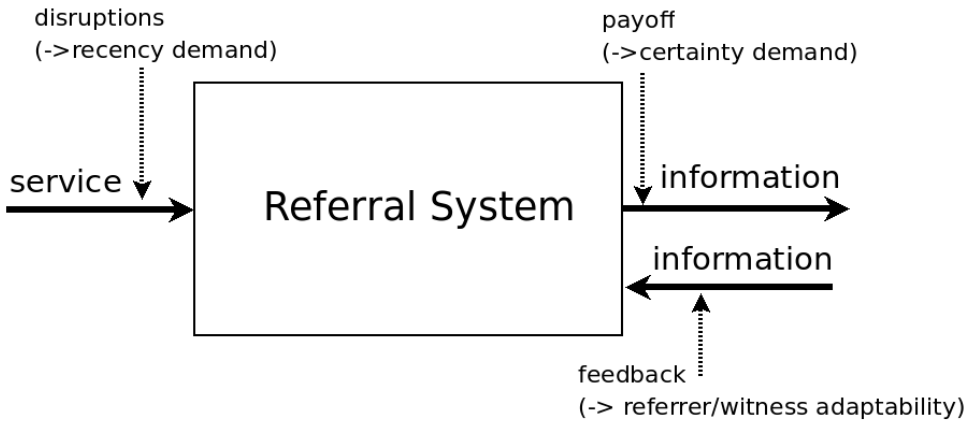


Figure 4.5.: Overview of the system

in quality are very regular. We propose to model a more natural pattern of disruptions which is also less predictable.In general, changes in the quality in service provision can be due to uncontrolled factors or due to a reaction by the provider on changes in the environment (for instance, decreasing usage or changes in reputation as in Huberman and Wu [2003]). We propose a model to model the former, since we want to make the disruptiveness of the service a controlled variable, much like Harrington Jr [1998] did. Our focus is on the agents in the referral systems (referrers and witnesses) and to them, the service is an environmental factor.
To model an uncertain, possibly disruptive, and non-deterministic service quality, we implement a service provider of whom we control the probability of service disruption (a change from good to bad quality or vice versa). The initial p is always 1.0 and gets updated according to the algorithm 4.2. The variable *disruptiveness* denotes how probable it is that the service quality changes.

Algorithm 4.2 computing probability of good service quality p

```
p ← 1
while running do
  if random() < disruptiveness then {switch service quality}
    p ← p * −1 + 1
  end if
end while
```

Figure 4.6 visualises the average probability of a disruption in 50 test runs among three settings, where *disruptiveness* was set to 0.05, 0.125 and 0.25.

4.3.2. Output: Client utility and risk behaviour

On the other end of the system, the client implementation should model that certainty actually has a value. The assumption that certainty is useful has to be set into practice.

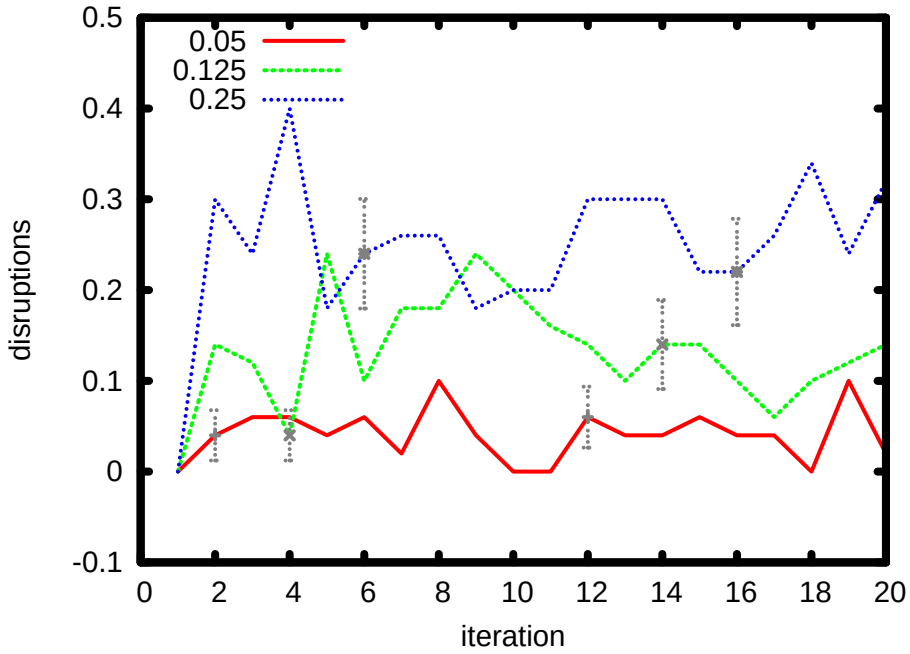


Figure 4.6.: Probability of service disruptions in test run

Therefore, we propose that the client can accumulate a payoff, representing the usefulness of the system to provide utility. The client takes a risk each round. He puts a stake into the interaction with the service provider. If the service is good, his stake gets doubled, otherwise he gets nothing back.
In our model, the amount of the stake relates to the belief of the referral network in a good service minus the disbelief. Thus, the stakes upper bound is the belief (if there is no disbelief) and the lower bound is zero (if disbelief is higher than belief). Stakes rise with certainty. So the more conflict is apparent in the trust network, the less risk the client will take.

Coming back to Luhmanns definition of confidence in the system, which enables personal and trust (see section 2.1), we can see how this simple model relates to his view. The clients translation from the observation of system confidence to a personal action builds a bridge between the clients confidence in the system and the personal (and risky) trust relation with the service provider. It is important to note that when we speak of payoff we do in fact consider the confidence and accuracy which the system is able to build up.

The algorithm listing 4.3 illustrates how the client proceeds, where *combined_trust* is the trust report gotten from the referral network after all concatenations and aggregations and *direct_experience* is the clients updated experience. Note that the belief of a trust report combines the estimated probability of a good service with the certainty.

Algorithm 4.3 A client updating its payoff and direct experience

```
stake ← max(0, combined_trust.belief() − combined_trust.disbelief())
if last_service == good then
    direct_experience.r ← direct_experience.r + 1
    payoff ← payoff + stake
else
    direct_experience.s ← direct_experience.s + 1
    payoff ← payoff − stake
end if
```

4.3.3. Input: Feedback from client

In later experiments, we make the referrers and witnesses act adaptively. This means that they control their own views of the world, first the trusts in their neighbours (which witnesses already in Hangs experiments from their first-hand experience) and later their own discounting behaviour. In Hangs model, referrers simply referred via a good or bad referral depending on the honesty of the referee. This is a severe simplification of reality. Referrers should learn themselves how trustworthy the referees are. For this, they need to receive feedback by the client for each of their referrals. With this feedback, they can use the update operator to update their own trust in their referees. Here, the referrer assumes that the referee is certain in his referral accuracy (see below for our revised version of the update operator).

4.4. The separation of trust accuracy and referral accuracy: Revisiting the update operator

To evaluate a referrer, we might judge him by two measures: The first measure would evaluate if he refers to witnesses who tell the truth, we call this *trust accuracy*. The second measure would be interested in weather the referrer can evaluate his referees himself accurately. He might happen to refer to a bad witness or referrer, but then a good referrer would know this and put a low trust in this path - we call this *referral accuracy*. In Hangs model, the update operator relies only on trust accuracy. However, we believe that referral accuracy needs to be considered, as it helps to assess which agent on a referral path actually accurately knows who he refers to.
Hangs update operator, described in section 2.3.3, takes into consideration the direct experience of the client and the trust which got concatenated along the referral path of the referrer (we could call this the opinion from his sub-network). However, during the application of the concatenation operator along the path, the actual opinion which the referrer had about this path becomes less observable. For example, if agent A refers to agent B who witnesses the service of C, concatenation will take place twice on this path. Let us now assume that either agent A or agent B has a very low opinion of the next agent in the path while the other one has a good opinion. After concatenation the end result will be the same (the witness report will be discounted once with a low belief and

once with a high belief). However, at this point it is hard to say who held the low belief and who held the high belief.
We therefore propose a different update operator here. We take three things into consideration:

1. the concatenated trust from the referrers sub-network *conc_trust*, which the client uses to make a decision
2. the actual opinion *act_trust*, which the client has of the service
3. and (in addition to Hang) the referral trust report ref_trust, which the client originally got from the referrer, reflecting the referrers opinion about his sub-network

We will first compute the the *trust accuracy* $\in [0, 1]$, which describes the usefulness of the path that the referrer provided and is still computed like described in section 2.3.3 using the concatenated trust tr and the direct experience td of the client:

$$trust\_accuracy = \frac{conc\_trust.\alpha^{act\_trust.r} * (1 - conc\_trust.\alpha)^{act\_trust.s}}{act\_trust.\alpha^{act\_trust.r} * (1 - act\_trust.\alpha)^{act\_trust.s}}$$

This is then compared with the referrers original referral ref_trust to compute the *referral accuracy* $\in [0, 1]$. What was the referrers opinion about the path? The difference between his opinion and the actual usefulness is the *referral accuracy*. When this difference is low, the referral accuracy is high:

$$referral\_accuracy = 1 - |ref\_trust.\alpha - trust\_accuracy|$$

The overall accuracy should contain both of these measures. Here, we simply weigh both equally:

$$accuracy = \frac{trust\_accuracy + referral\_accuracy}{2}$$

The client can now update his trust in the referrer $tref$ in the usual way:

$$tref.r = ref\_trust.certainty * accuracy + (1 - \beta) * tref.r$$

$$tref.s = ref\_trust.certainty * (1 - accuracy) + (1 - \beta) * tref.s$$

This method stops laying all weight on who the referrer knows and incorporates if he can evaluate the paths he provides correctly. As another example, consider the case where a referrer refers to a bad witness (who gives a report contradicting the actual experience). However strongly the referrer discounts the report with his belief, this doesn't change the α of the path (see also our discussion on the concatenation operator). Hangs operator would evaluate the referrer negatively even if he put almost zero belief in the witness.
The approach we propose can distinct between both abilities of referrers and let them both equally weigh in. Future research may try different settings for the weight between the two notions. We note that referral accuracy is especially of importance when the network becomes more dynamic and agents adjust behaviour more often or enter and leave the network frequently.

4.5. Discounting strategies and personal history

In Hangs model, only the client had a discounting strategy, because the focus of interest was in how the client can model the referrers using the update operator. However, the client could, in addition, also discount his own trust in the service provider. Other agents (witnesses and -later- adaptive referrers that do keep an own history) might also discount their memory. Our model includes all these discounting strategies.
Of course, it needs some consideration how different discounting strategies affect such referral networks (see our hypotheses in chapter 3). We will start by looking at one of Hangs experiments again in the experiment chapter.

Agents should be able to determine their discount rate freely. When referrals are accurate, it seems worthwhile to have a low β in the hope of accumulating certainty. When they are way off, β should go up in order to avoid further losses - presumably, the old information is worthless.

For the client, we propose to tie his discounting factor β to the trust accuracy his network delivered in the last run (see the preceding section on how trust accuracy is computed):

$$\beta = 1 - trust\_accuracy$$

Referrers and witnesses can alter their discounting factor β while receiving feedback about their referrals. To this end, we make each agent keep a trust report about his accuracy history, updated by feedback. (using the update operator). This trust in the own history monitors how, from the agents perspective, referrals made from his point of view matched reality. We can regard this personal history as learning about the world. To use a personal history was also proposed in Wang [2009]. The agent can then update β by using the α of the trust in his personal history. When referrals were good, β should be low in order to gain certainty. If they were bad, the history indicates that β should be low because old knowledge might have been misleading.

$$\beta = 1 - trust\_in\_history.\alpha$$

Lastly, each agent needs to make sure that all his trusts he holds will from here on get discounted by the new β. An interesting topic is also what the β of the personal history should be. By setting different values for this, the agent varies the influence of the past on how much memory he keeps for referrals.

4.6. Path selection / Cycle detection

In order to avoid cycles in referral paths, we make the client check before he added an agent to the referral tree if that agent had already been referred to on the current path along the tree. If so, he stops following that path. Then, this path might be a dead end, so the client retracts it upwards until it finds a branching (i.e. the next parent is used to refer to more than one node and is thus still useful).

In the setup of Hang et al. [2008], there were many witnesses compared to referrers, making it likely that paths were short. This measure avoids cycles.
In our case, we need a better solution, because these conditions would not hold in all of our experiments. We want to investigate effects of small and large values of the neighbourhood size parameter k and the ratio between witnesses and referrers l.

Another approach would be to keep a list of all visited nodes in order not to visit any node twice. This would have the effect of providing a partial solution to the path selection problem we explained shortly in section 2.3.3.4. Double counting would be avoided because the client stopped following a path when he would visit a node twice. Of course, this solution would be partial because it is not sensitive to which path would have been more desirable to follow. An idea would be to order the list of referrers before collecting their referrals, which would implement a simple selection strategy.

5. Experiments

In this section, we describe experiments we conduct in terms of setup and results. We start with explorative scenarios for parameter analysis of our model and then conduct adaptive scenarios.

We run experiments in three phases. As has been said in chapter 4, we recreated the experiments by Hang et al. [2008] in a *consolidation phase*, in which we expanded the simulation code and used these experiments as integration test cases (for more cases see the appendix). Here, we conduct an *exploration phase*, in which we introduce our disturbance/risk model and explore the settings concerning the honesty of agents.
Furthermore, in the *adaptation phase*, we add more adaptiveness to the referral system - referrers will control the trust they have in their neighbours and all agents get to control their discounting strategy.

5.1. Design

Tables 5.1 and 5.2 show the dependent and independent variables for experiments from our exploration and adaptation phase. We run experiments with $2^k factorial$ design.
Our simulations are longer than Hangs experiments (50 epochs), to observe effects with longer reaction times.
Each experiment setting ran 50 times and we show the averages along with the sample standard deviations. In these scenarios, all settings are set for the whole run of an experiment. Exceptions are noted in the setup description. We number our experiments starting with four, to keep the numbers one through three denoting Hangs experiments. We refer to the hypotheses made in section 3.2.2 when the results indicate that they can be accepted or refuted. We conduct statistical T-tests when the difference between outcomes is crucial to the discussion.

Name	**Measured In**
Client trust in an honest referrer (α and certainty)	3a
Client payoff	4a, 4b, 5, 6, 7, 8, 9
Trust among referrers (α and certainty)	7
β in referrers and witnesses (values and standard deviation in single runs)	9

Table 5.1.: Dependent variables

Name	**Possible Values**	**Varied In**
Epochs	**50**	-
N (number of referrers + witnesses)	**18**, 24	6
l (ratio of witnesses to referrers)	0.2, 0.5, **0.8**, 1.0	6
k (neighbourhood size of client)	**0.4**	
k (neighbourhood size of referrers and witnesses)	0.2, **0.3**, 0.5, 0.8	6
Provider type	damping, **disruptive**	4a
Probability of service disruptions	0.05, **0.125**, 0.25, 0.33	4b, 7, 8, 9
β (discounting factor) of client	0.0, **0.4**, 0.8	4a
β of witnesses	0.0, **0.4**, 0.8	3a, 4a, 4b
Number of bad referrers	**0**, 4, 8	5
Number of bad witnesses	**0**, 1, 4, 7	5, 7
Referrers adapt trust in neighbours	**no**, yes	7, 8, 9
β of (adaptive) referrers	0.2, **0.4**, 0.5, 0.8, random	8
Referrers and witnesses adapt β	**no**, yes	9

Table 5.2.: Independent variables and which experiments vary them. When they are not varied, the (bold) default value is used.

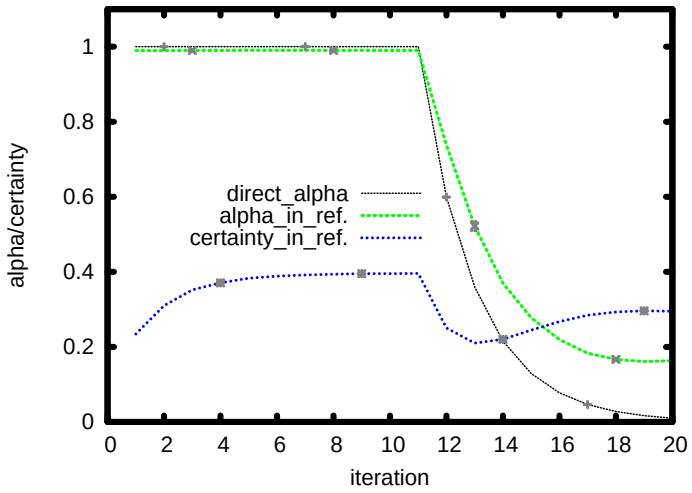
5.2. Exploration

5.2.1. Experiment 3a: Discounting Alignment

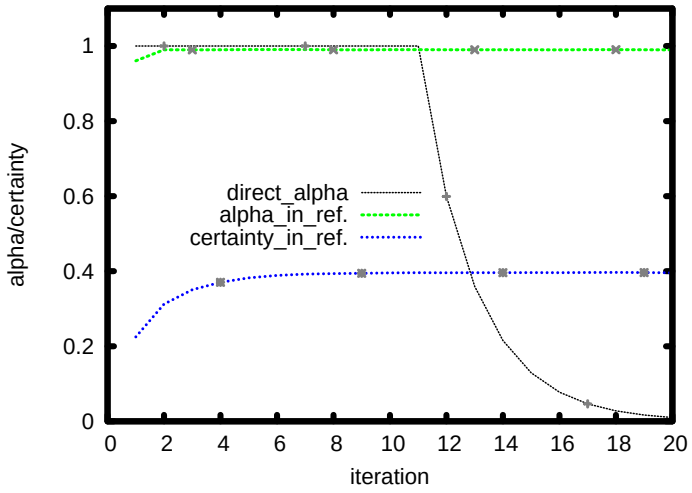
We remodel the third experiment of Hang et al and make the witness use different discounting factors.

Setup It is unclear to us if the witnesses in Hang et al. [2008] used any discounting at all for their trust in the provider. However, when we tried different values for the witnesses β in experiment 3 (where the client used $\beta = 0.4$ for his trust in the service provider), we found that the results changed significantly.

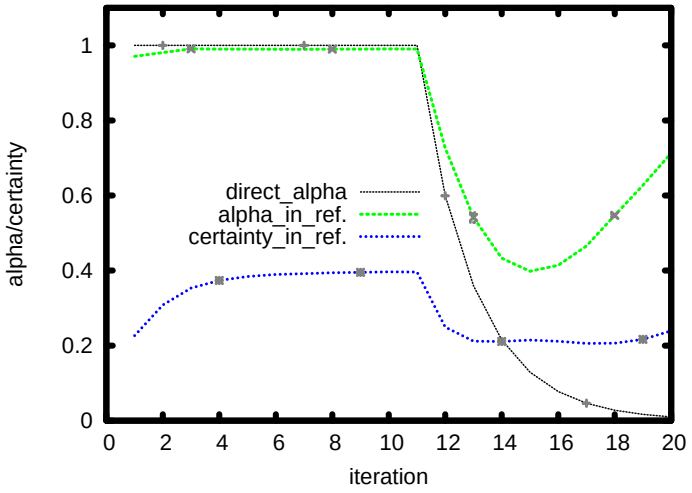
Results The trust in the honest referrer stabilises more quickly after the service disruption when the client and the witness are discounting their trust in the service provider



(a) witness $\beta = 0.0$



(b) witness $\beta = 0.4$



(c) witness $\beta = 0.8$

Figure 5.1.: Experiment 3 with differing values of β for the witnesses

at the same rate ($\beta = 0.4$). Figure 5.2.1 shows the results for Hangs experiment 3 with varying β for the witnesses. Clearly, with the witnesses using $\beta = 0.0$, the client has a hard time adjusting his trust in the honest referrer, since a lot of old information still flows through the network. With the witnesses using $\beta = 0.8$, information flowing through the network is newer than what the clients memory of the service reflects, so it takes time until he updates his trust in the referrers correctly.
We conclude that hypotheses $4a$ **seems to hold**: differing discounting strategies matter for reactiveness. However, **hypothesis** $4b$ **does not seem to hold**: it is beneficial if agents use the same β. It is an issue in need of investigation, however, if it can be generally assumed that aligned discounting factors will result in the best performance. For now, we indicate that it might be harmful for a referral system if agents use different discounting strategies, i.e. when their memories are out of sync. We will return to this question in the adaptation phase (experiment 8).

5.2.2. Experiment 4a and 4b: Disruption/Risk-Model

In this experiment, we are interested in the effects for client utility when trying out different values for the service quality disruptiveness and the recency of information flowing through the network.

Setup All referrers (10) and witnesses (8) are honest. We vary the disruptiveness of service quality changes and the discount factor which the client and all the witnesses use. Due to the lessons from experiment 3a, the client discounts the trust in the service provider with the same β as the witnesses, so in this experiment we control how old all information flowing through the network is, ruling out any conflicts due to differing discount rates among agents.
In experiment 4a, the service provider is damping, just like in Hangs experiment 3 and we are interested in the effect on performance. In experiment 4b, we actually employ the service provider we sketched in section 4.3, whose service quality changes abruptly, given the disruptiveness setting.

Results Figure 5.2 shows how discounting affects performance when the service suddenly drops. This service provider is very predictable before and after the only disruption, so the setting $\beta = 0.0$ is generally the most profitable. However, while having $\beta = 0.4$ or $\beta = 0.8$ provides stability immediately after the disruption, $\beta = 0.0$ makes it hard to provide accurate judgements for some while after a server disruption.

Figure 5.3 shows results with the disruptive service provider. We note how using a low discount factor ($\beta = 0.0$) is highly profitable when the service quality is stable (just as in experiment 4a), as it can rely on more certainty and risk higher stakes. But this is less and less the case as the service becomes more disruptive (as the disruptiveness factor increases). With a disruptiveness factor of 0.25, it becomes more useful to use a higher β, like 0.4. When service disruptions are more common, the profits of $\beta = 0.4$ surpass those of $\beta = 0.0$. This indicates that for a given uncertainty in the environment (here: a

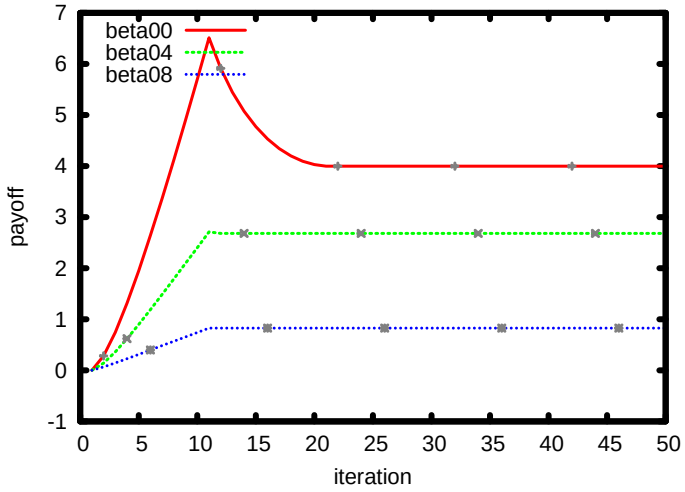


Figure 5.2.: Results of Experiment 4a

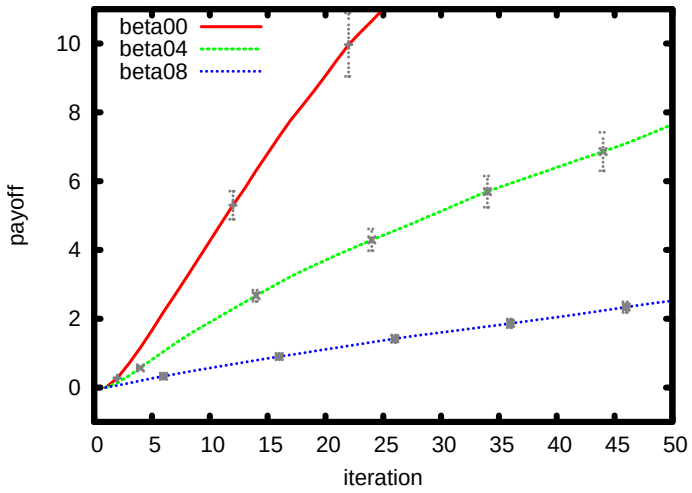
given disruptiveness factor), there is an optimal discounting factor. **The observations validate our second set of hypotheses** ($2a$: The environment is highly influential on the success of discounting strategies; $2b$: Low discounting is hurtful in disruptive scenarios). We also not that the variance is relatively high with $\beta = 0.0$ under high disruptions, i.e. the payoff becomes less predictable.

5.2.3. Experiment 5: Populations with mixed honesty

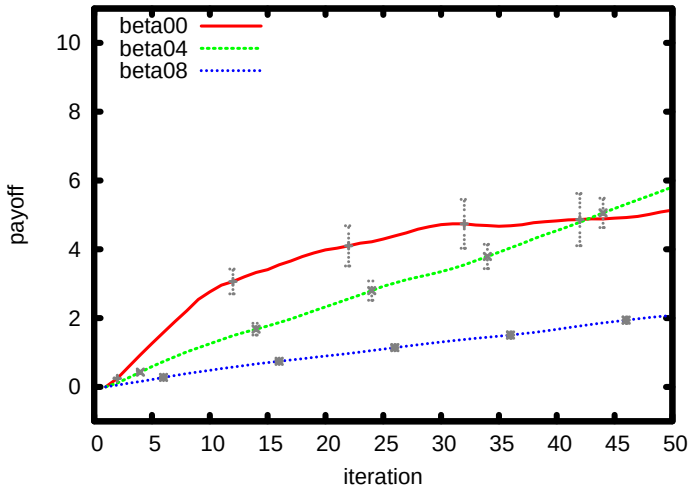
Here, we introduce some dishonest referrers and witnesses and observe the effect on the system utility (the clients payoff). We do this with non-adaptive referrers since we do not model dishonest adaptive referrers.

Setup As usual, the client and witnesses use $\beta = 0.4$ and the service disruptiveness fixed to 0.125. We run experiments with 8, 4 and 0 bad referrers and 7, 4 and 1 bad witnesses.

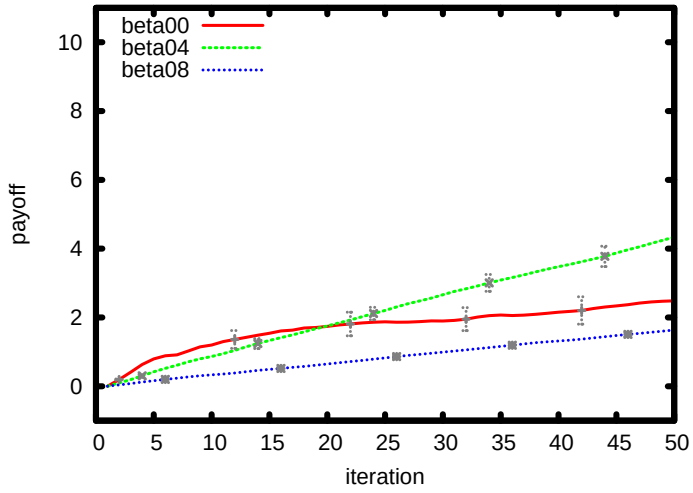
Results Figure 5.4 shows the payoffs for runs with 8, 4 and 0 bad referrers. In each plot, we see the graphs for 1, 4 and 7 bad witnesses. If we compare these payoffs with the setting from experiment 4b, in which all agents were honest, we note how wrong information hurts certainty and thus the payoff. This is by design as conflict in the information decreases certainty (see section 2.2).
For both bad referrers and bad witnesses holds: if there are too many of them, the system generates significantly less payoff. If all witnesses but one are bad, the system makes no profit (and only when only 2 referrers are bad can a loss be avoided). If 8 out of 10 referrers are bad, the system can make a profit, provided there are enough good witnesses. Interestingly, both factors can cancel out the bad effects of the other. For example, consider the runs with 4 bad witnesses (green line). It takes more than 2 good referrers in the system to make a profit. On the other hand, if there are 8 bad referrers, it takes almost all witnesses (7) to be good to make a profit.



(a) disruptiveness = 0.05

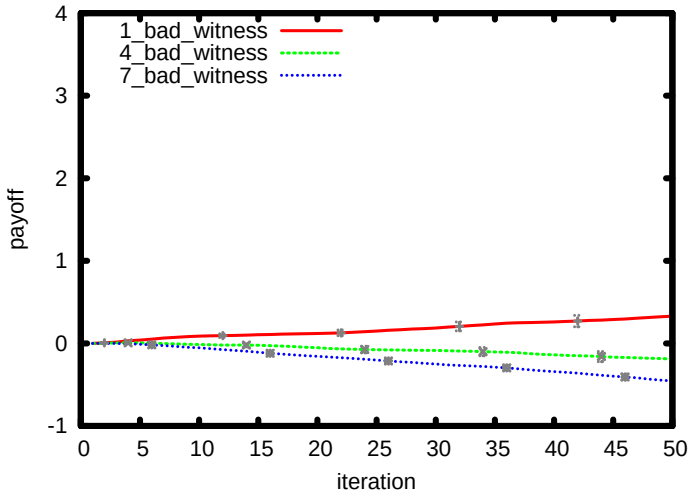


(b) disruptiveness = 0.125

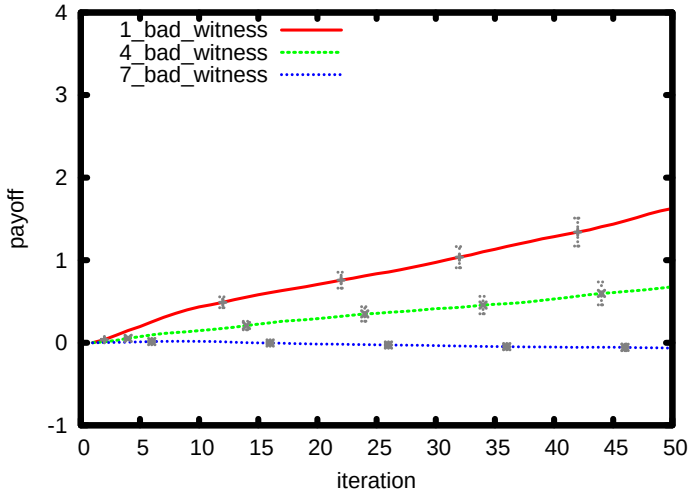


(c) disruptiveness = 0.25

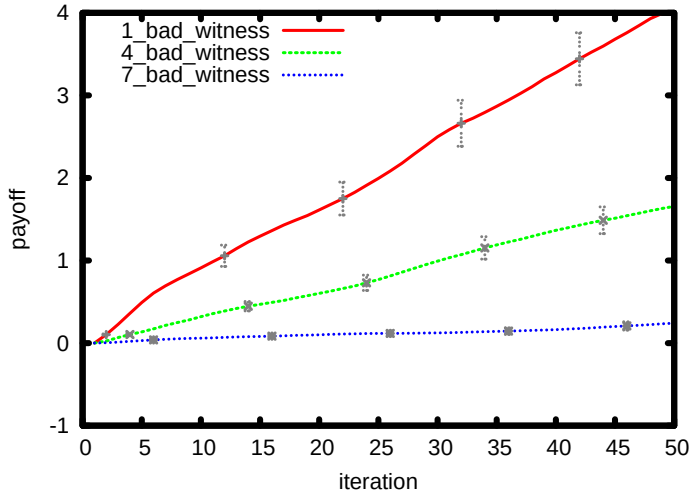
Figure 5.3.: Results of Experiment 4b



(a) 8 bad referrers



(b) 5 bad referrers



(c) 2 bad referrers

Figure 5.4.: Results of Experiments 5

This experiment also tests for the systems capability to lay more weight on true information. The client updates his trust in referrers and thus weighs how much he listens to the opinions they provide. **We conclude that hypothesis** $1a$ **holds**: high percentages of dishonest information can be tolerated by this system (although it hurts performance, but losses were really low).

5.3. Adaptation

In a further set of experiments, we added adaptive behaviour to the agents. To this end, we implemented a feedback from the client to the referrers about the usefulness of their referral (see section 4.3.3). Referrers are equipped with their own trust table for each referrer or witness they refer to. They update these trusts after receiving feedback. Note that referrers still chose randomly which agent they will refer to (with a preference to refer to witnesses). It would make sense to refer to the agent they trust most, but for now this stabilises the system too quickly and makes comparisons harder.
We will also enable agents to adjust their discounting factor β autonomically.

5.3.1. Experiment 6: Adaptive Referrers

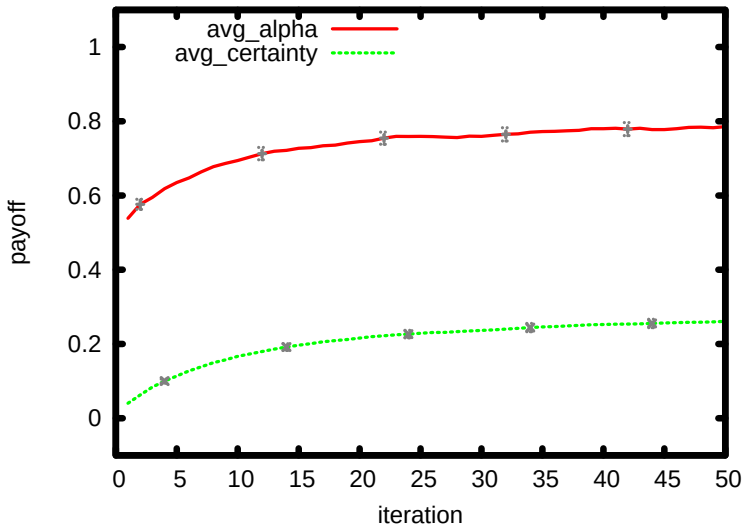
The initial experiment in which referrers are finally made more adaptive. Until now, they simply relayed with a very positive or very negative trust, relying on global knowledge if the referree is generally honest. Now they start with an empty trust in all their neighbours and update that when the client gives feedback about the referrasl (see chapter 4).

Setup The discounting factor of all agents is set to 0.4. There are three scenarios, in which we inject 1, 4 and 7 bad witnesses, respectively.

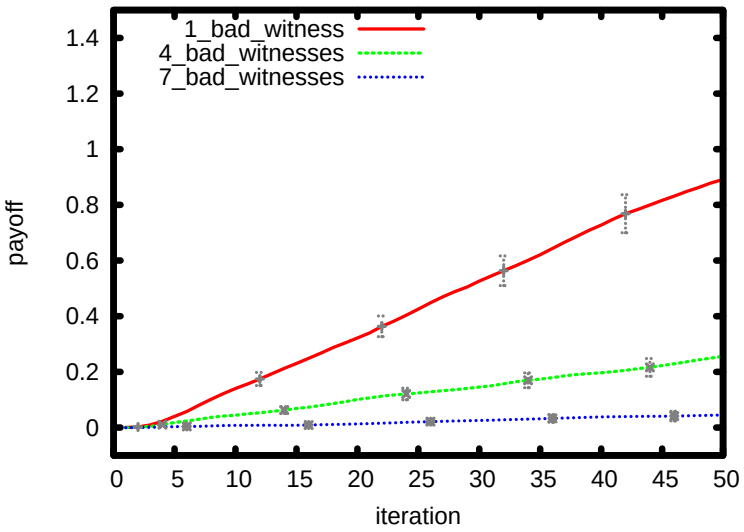
Results Figure 5.5 shows that the trust among referrers is well. However, certainty does not rise above 0.3, due to discounting along the paths when trust reports get concatenated. This influences the risk behaviour of the client, who earns less payoff (note how the scale of the plots have changed). In a similar scenario of experiment 4, his payoff rises to over 6.0 after 50 iterations.
However, in all previous experiments, referrers are nothing more than relays, attaching very positive or very negative trust to all referrals. Now, with autonomously learned trust in an unstable environment, trusts will be less extreme and the system therefore less certain. The worlds we model in previous experiments are artificial in that respect.

5.3.2. Experiment 7: Structure

We vary the number of referrers each referrer has as neighbours. In addition, we are interested in the effect of the ratio between witnesses and referrers.



(a) Trust among referrers with 1 bad witness present: average alpha and certainty



(b) Payoffs with 1, 4 and 7 bad witnesses

Figure 5.5.: Results of Experiments 6

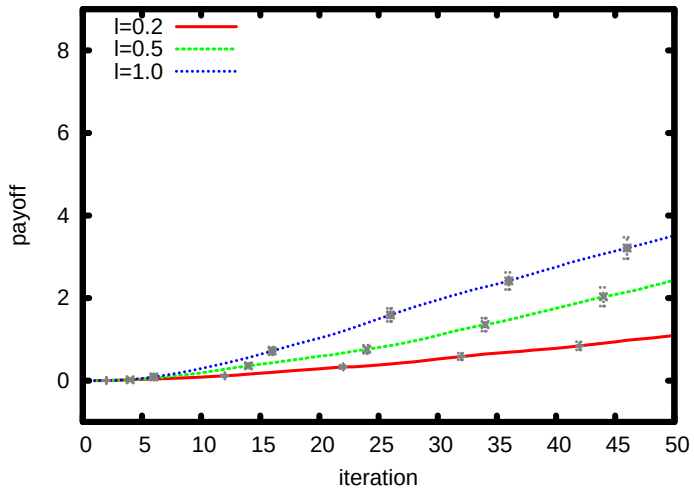
Setup The model is run with different connectivity settings. We vary the neighbourhood size k for referrers, which denotes the ratio of referrers known to a referrer, and l, the ratio of witnesses to referrers present in the system. Note that the system in HANG ET AL. [2008] had 10 referrers - the client knows 4 of them, so Hang uses a k of 0.4 for the client. Referrers have two referrers as neighbours, so they have a k of 0.2. He also uses 8 witnesses, which would denote his l at 0.8. These are also the settings we normally use.
For this experiment, we increase the number of agents employed to 24 (the client and the service provider not counted) to make connectivity effects clearer. We use the values 0.2, 0.5 and 0.8 for k (which translates, to different actual rounded values, depending on l) and 0.2, 0.5 and1.0 for l (which translates to 4, 8 and 12 witnesses present, respectively). For instance, if $l = 0.5$, we have 8 witnesses and 16 referrers. If $k = 0.2$, then each client and referrer has $16 * 0.2 = 3.2 \approx 3$ referrers as neighbours.
In general, one would expect the referral paths to become longer with decreasing k or l, since this makes it less likely for a path to a witness to be completed with few referrals. Longer referral path should lead to lower certainty, as beliefs are concatenated and multiplied.

Results We note how l does affect the performance. If there are more witnesses present, performance rises. We looked at average path length and confirmed that paths get longer with smaller values for k (in a range between 2.2 and 4.2). Longer paths indeed mean reduced certainty and thus reduced performance.
Interestingly, k has no significant effect. This is due to the random routing among referrers - referrers have no preference who to refer to when they know no witness. This could change if smarter routing strategies are employed - we assume that it is beneficial to refer to agents whose connections promise a shorter path.
The results **confirm our hypothesis** 1*b* (structure affects performance). However, the effects of k are clearly not as strong as the honesty effects from experiment 5. This leads us to believe that the information quality is the more important factor for our system.

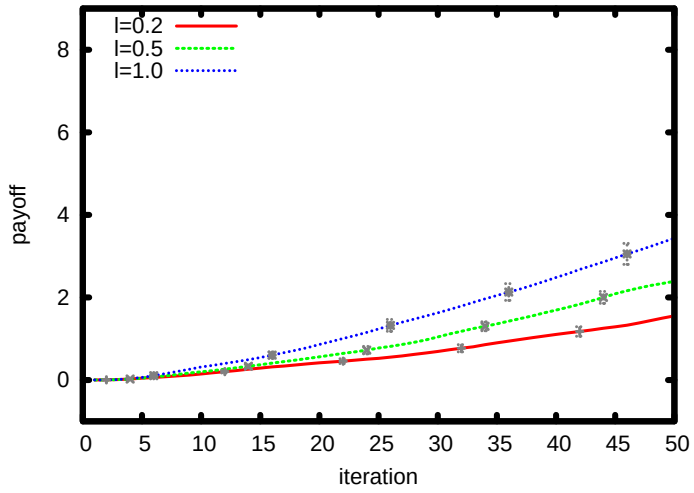
5.3.3. Experiment 8: Discounting Distributions

The same general amount of discounting in the system is employed among the client, referrers and witnesses in different ways and the effect on these scenarios on the system performance is observed.

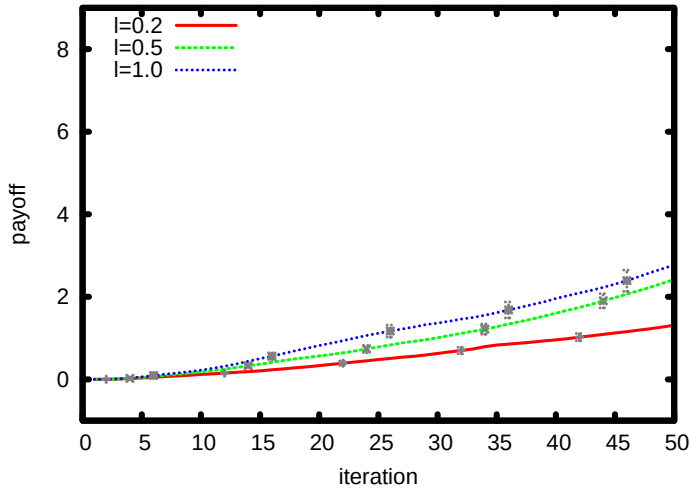
Setup Three scenarios model different distributions of discounting variables among referrers and witnesses. It is important to notice that the same amount of discounting will be in the system, only distributed more or less evenly among the agents. We chose 0.5 for this amount. In one scenario, all agents will use $\beta = 0.5$, in the second all βs will be random $\in [0, 1]$. In the third scenario, each agent is equally likely to be assigned $\beta = 0.35$, $\beta = 0.5$, or $\beta = 0.65$.
We set the disruptiveness to 0.33 in this experiment, in order to create an environment



(a) $k = 0.2$



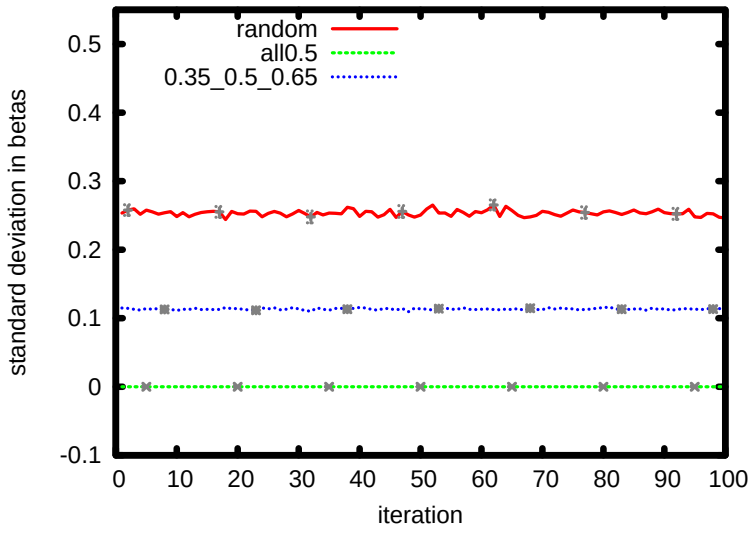
(b) $k = 0.5$



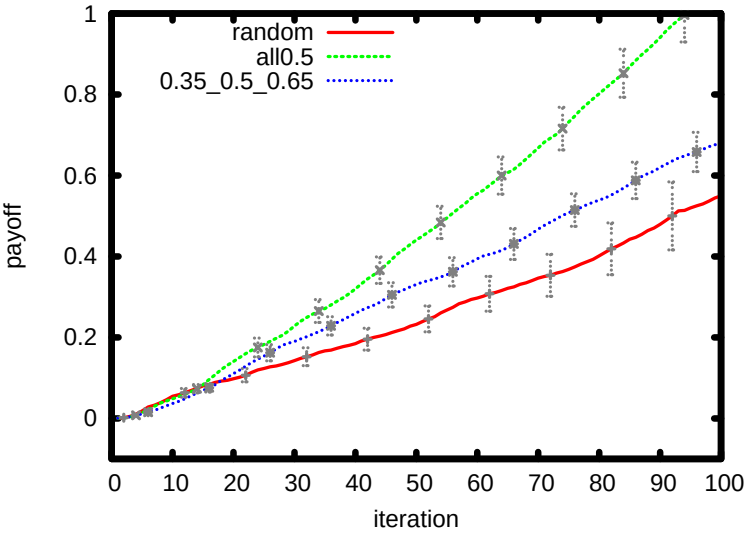
(c) $k = 0.8$

Figure 5.6.: Results of Experiments 7

where 0.5 is a reasonable value for β. Note that from now on, we let the experiments run 100 iterations, in order to observe effects with time dynamics.



(a) The standard deviations among values for β



(b) Payoffs

Figure 5.7.: Results of Experiments 8

Results In figure 5.7 we can see the different deviation levels in the values for β for each scenario (note that there are still fluctuations as we only record the agents that took part in the referral tree which the client builds in the current round). There are of course no deviations when all agents use $\beta = 0.5$, some more when they use three different values and about 0.3 when β values are completely random. Coming back to our intuition from experiment 3b, **we can see that hypothesis** *4b* **does not hold**: The scenarios with

small deviations in β among the agents are more successful. Also, they have less deviations.
In order to test for the significance of the results, we conducted Welsh Two-Sample Tests considering the last iteration to test for the difference between the payoff in the scenario where all agents use 0.5 and the (mixed) scenario in which agents us three different values ($t = 4.4969$, $df = 88.493$, $p - value = 2.083e - 05$, mean performances were 1.055 and 0.675) and between the mixed scenario and the random one ($t = 1.2066$, $df = 72.438$, $p - value = 0.2315$, mean performances were 0.6758 and 0.5446).
We attribute these significant effects to the higher probability of referral paths consisting of agents who use the same β.

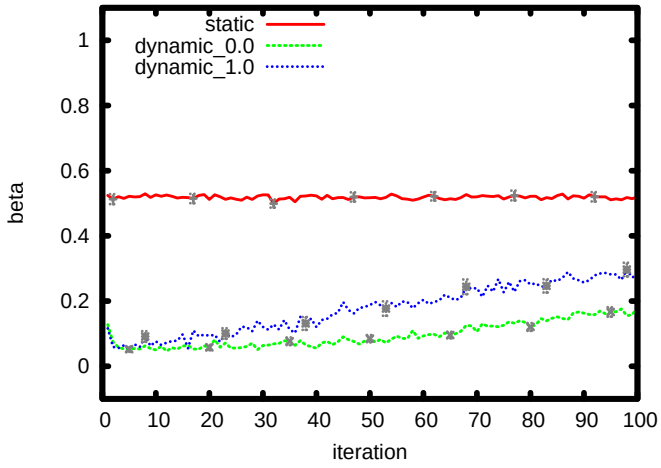
5.3.4. Experiment 9: Adaptive Discounting

Agents adapt their discounting factor β using the accuracy of their referrals.

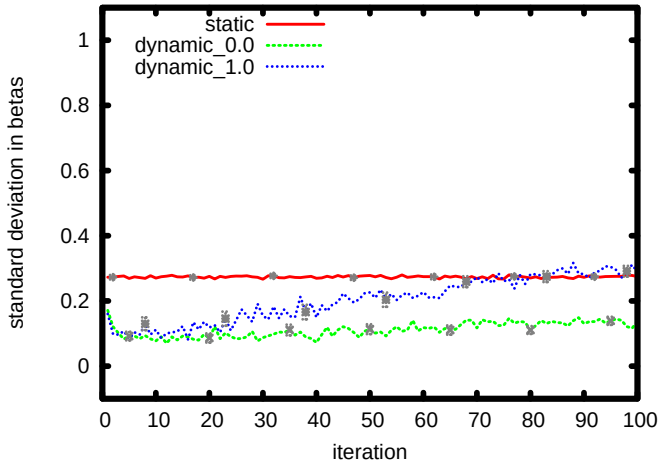
Setup All β values are initialised randomly $\in [0, 1]$. In the static scenario, all β values will stay that way during the runs. In the dynamic scenarios, the client, the referrers and the witnesses are enabled to adjust their β (see section 4.5 for technical details). They use a trust in their own history, of which the β (which we will call β_p here) is adjusted for in three scenarios (Here, high values mean that the agent stores a history of how accurate things were): It is either set to 0.0 (meaning that all history of accuracy is kept) or 1.0 (meaning that no history is kept).

Results In figure 5.8, we see that payoff-wise, the dynamic scenarios fare much better than the static scenario. **We confirm that hypothesis** 3 **holds**: It is beneficial to adjust β values according to the situation at hand.
If we look at the actual values of β values, we see that the static agents stay at an average β of 0.5 with a standard deviation at around 0.3, which is expected with a random assignment. We also see that agents in the dynamic scenarios choose lower β values around 0.2 with a standard deviation at around 0.1. In fact, the graphs of single runs show very low β values and short spikes whenever disruptions in the service happen.
So what might actually lead to improved performance? This can have two reasons: First, the agents might have found a better β for the given uncertainty in the environment. It makes sense to forget quickly when the circumstances changed (thus the spikes) and to build up certainty in stable times. Second, assume the agents settle on some β due to the algorithm we chose, but let us disregard if it makes any difference on the system performance. Still, the lower deviation among the values might benefit the system performance (which would relate to our findings from previous experiments and hypothesis $4a$).
Another observation on the system level is that agents do equally well, performance-wise, with $\beta_p = 0.0$ or $\beta_p = 1.0$ for the personal history. They do not seem to profit from the ability to generalise over the past. However, with $\beta_p = 0.0$, deviations in β values are significantly lower. We conducted Welsh Two-Sample Tests to test for differences when using $\beta_p = 1.0$ and $\beta_p = 0.0$: in β values ($t = 5.0917$, $df = 88.004$, $p-value = 1.999e-06$,

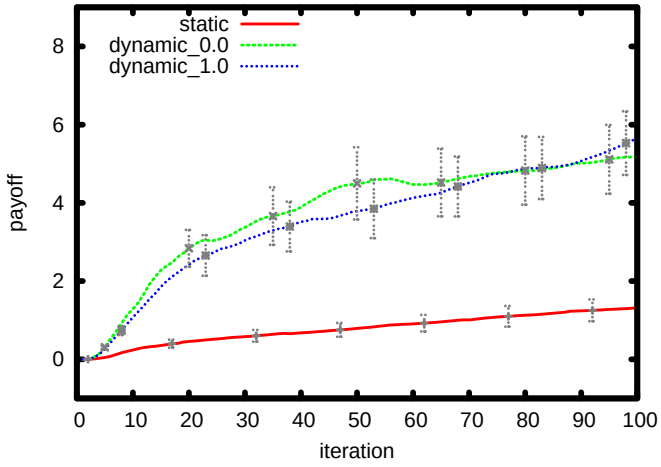
mean beta were 0.272 and 0.157) and between the deviations ($t = 11.074$, $df = 97.204$, $p - value = 2.2e - 16$, mean deviations were 0.31 and 0.11). The differences are both significant. This observation indicates that agents who remember their personal history ($\beta_p = 0.0$) change their β values less abruptly, notably without hurting performance. This approach seems more natural and might prove beneficial for the agents in more complex settings.
Finally, we note how standard deviations of the general β values among agents tend to rise with the lifetime of the system. This effect is a hint that complex dynamics lie in this setup, yet to be discovered and explained. It could be that agents react to changes differently, according to their role or position in the network and differences build up.



(a) Avg. values for β among referrers and witnesses with disruptiveness = 0.125



(b) The standard deviations among values for β with disruptiveness = 0.125



(c) Payoffs

Figure 5.8.: Results of Experiments 9

6. Discussion

6.1. Contributions

In this work, we worked on the trade-off problem between certainty and recency in information systems. We established a testbed for certainty-based referral trust. We took the work by Hang et al. [2008] and extended it. Among other contributions, we discussed a new way of looking at the update operator, defined behaviours for the service provider and the client with which makes analyses of disruptive environments possible and implemented strategies to update the discounting rate by agents autonomously.
We conducted a series of experiments to explore the influence of several parameters. Finally, we made all agents adapt their trust in one another and let them adjust their discounting rate (the rate of forgetting).

6.2. Conclusions

While Hang et al have shown that a trust network in which agents use certainty-based trust is reactive, we were interested in performance. We defined performance in our model as the ability to generate truthful and certain information as output, on behalf of which a client can make successful decisions about his interactions with the environment.
The results from our experiments showed us that such a referral system can be generating truthful and certain information in a stable way. We also learned that the amount of honest information has more impact on the performance than structural parameters. Of course, this might change if strategies are employed which target the structure of the network and use more intelligent routing.
More importantly, our findings suggest that the distribution of discounting strategies among agents is a crucial factor for reactiveness and performance. Even if the same overall amount of discounting takes place in the whole system, the variance in applied strategies can make significant differences. We presented simple strategies for agents to choose their discounting strategy freely, and saw that the system can generate significantly more performance this way. Patterns of discounting factors developed more naturally (with less deviations) when agents collected experience about their previous accuracy in a personal history. From our experiments, we expect complex dynamics at play when discounting factors are defined locally by all agents.

6.3. Future Work

In closing, we propose several research ideas which might lead to further insights:

Discounting Of course, the local strategies with which to choose the discount factor can be developed further: For instance, should agents be more forgiving? Or would it be a useful approach to implement a swarming behaviour where agents use a discounting factor which is similar to what their neighbours use? Basically, any machine learning algorithm could be tried, but we note that finding the best discounting factor is not an easy task: The environment is a moving target and feedback is supplied by third parties (if you are not a client), who themselves relied on network input. Of course, this also means that it is an interesting problem.

Structure Regarding the network-structure, one could try to use several clients and/or several service providers in order to analyse what this means for the flow dynamics in the system. One could also give agents more freedom to decide who to refer to. There are many simple heuristics to be tried - the first of which is of course to refer to the agent(s) one trusts the most (but caution has to be taken not to reduce the set of choices each agents considers by this). One more structure-wise approach is to let agents enter and leave the network and watch adaptability as trusts have to be built anew. A hard problem in many trust networks called "whitewashing" (Feldman et al. [2006])) is that agents can leave and come back with a new identity and a clean slate. A certainty-based trust representation might be suited well to tackle this problem.

Strategies Of course, there are strategy considerations. One could let clients acvtively make use of their own direct experience with the provider, which we left out. We made a distinction between trust accuracy and referral accuracy, but simply weighed both equally. One could try to use different weights for these concepts when agents update their trusts. Referral accuracy might prove worthwhile to maintain certainty in the trust towards agents when changes in the population happen often. It would also make sense to add incentives for referrers and witnesses. Like in the ART testbed (Fullam et al. [2005]), agents could be paid for referral services (this would also work well with heuristics in routing, since agents that are seldom asked for referrals could lower their price).

Our preferred scenario for further research would have clients share parts of their generated utility as payments for referrals and let agents leave (for instance if they become too poor) and return. This scenario would open many challenges and be of interest for real-world systems.

A. Appendix

A.1. Resources

The program code for running the described experiments can all be found online at HTTP://WWW.ASSEMBLA.COM/SPACES/TRUSTCERTPROP.

You can find a code browser there and a subversion code repository at HTTPS://SUBVERSION.ASSEMBLA.COM/SVN/TRUSTCERTPROP that can be checked out anonymously.

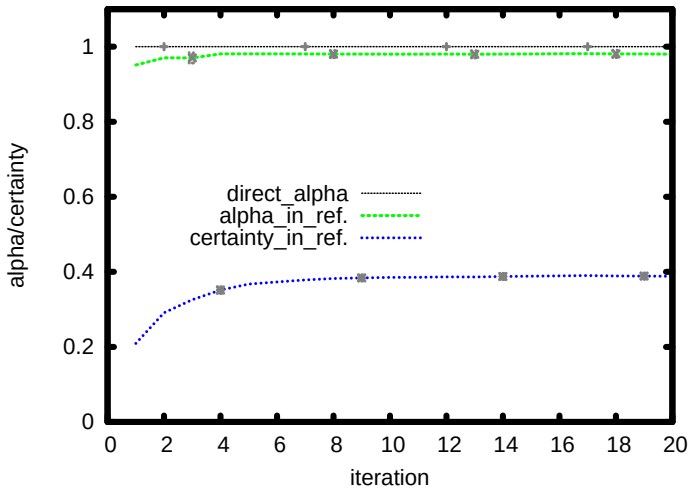
Note that the experiments need an experiment running suite in order to be executed and to create graphs. This was developed by Nicolas Höning and can be found at HTTP://WWW.ASSEMBLA.COM/SPACES/COMBEX.

A.2. Integration Testing in complex system development

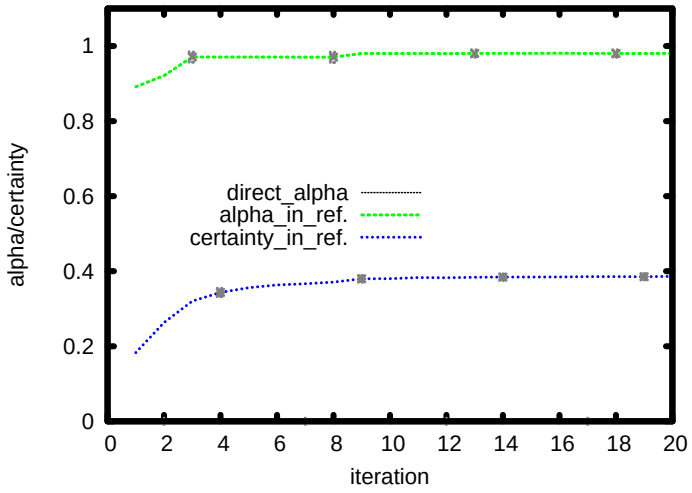
When working on a complex multi-agent system, we find it to be quite challenging to develop features and predict for every change which consequences it might have. This is especially true if a system contains certain randomisations. We believe it to be good practice for complex system developers to use simple scenarios (which test a combination of settings whose results are clearly predictable) for integration tests (Beizer [2003]). When the graphs look as expected, we can be more confident that our change did not disrupt any expected system behaviour.
For example, we found that the idea of stability versus complete disruption, present in in experiment 3 by Hang, provides good test cases for our system.

As first test cases, we conduct three extension cases of experiment 3, where the service provider either stays good throughout the whole run, starts bad and then upgrade to providing good service after 10 runs or starts bad and stays bad throughout the whole run. Figure A.1 shows those extra three scenarios we ran for experiment 3b and their (expected) behaviour. During development, they helped in finding several problems with referral tree building.

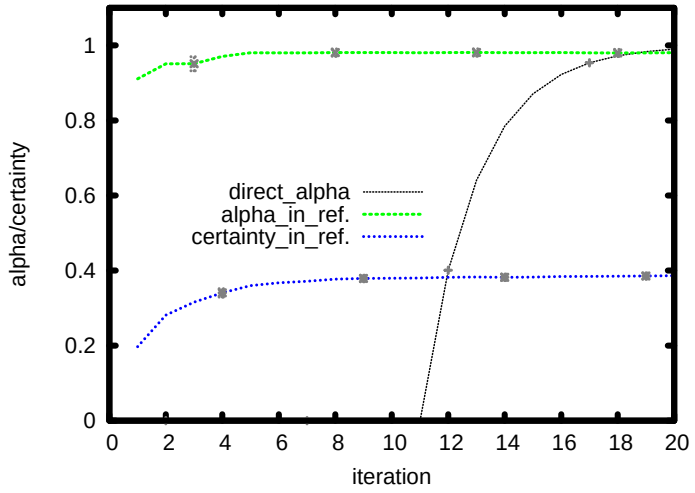
Furthermore, we were interested in the effect on trust in referrers when all available information was bad. We included in our test suite a scenario like the one before, the only difference being that all witnesses were dishonest and referrers adapted their own trusts in their neighbours. Figures A.2 and A.3 show the results. Note how trust in referrers is bound to about 0.5, which corresponds to their ability to refer adequately.



(a) Continuously good service



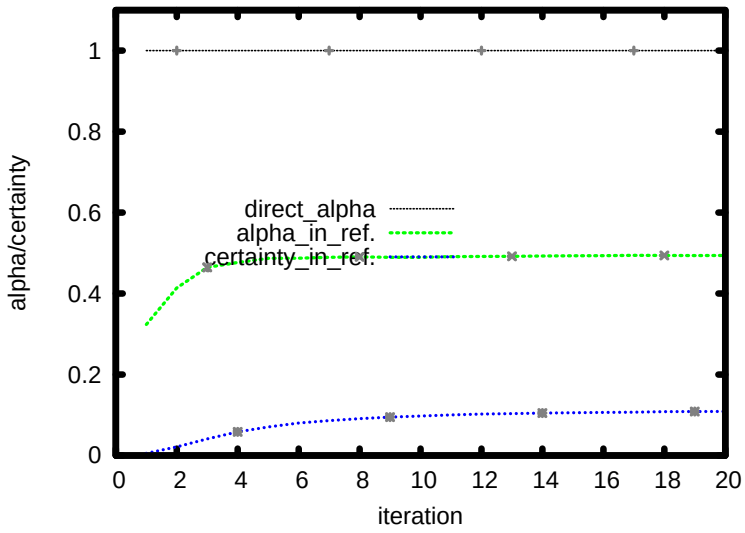
(b) Continuously bad service



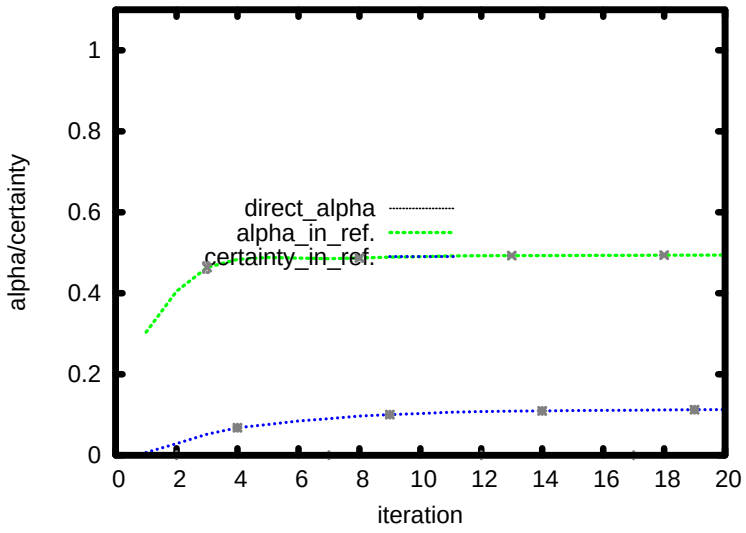
(c) Service starts bad, then upgrades

Figure A.1.: Results of Integration Test using Experiment 3

The other 0.5 are missing since they don't know any credible sources. See our discussion about the update operator in section 4.4.

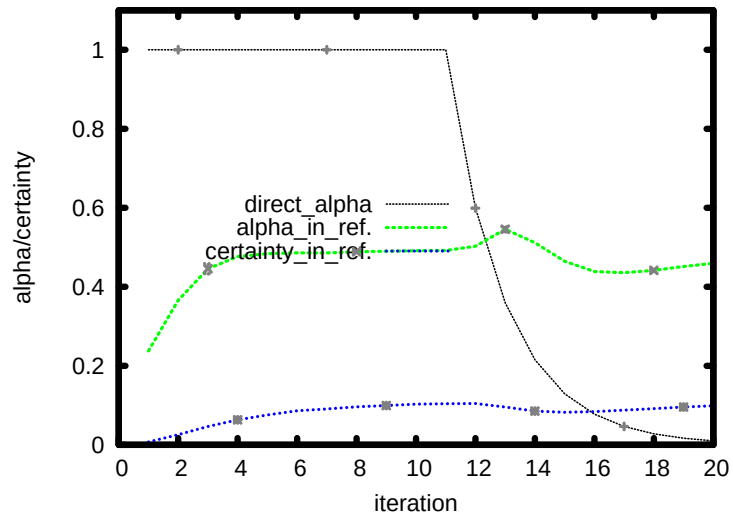


(a) Continuously good service

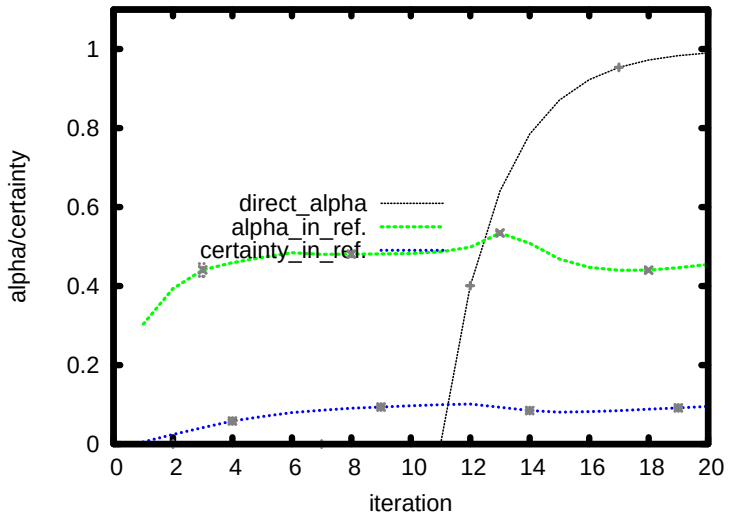


(b) Continuously bad service

Figure A.2.: Results of Integration Test using Experiment 3 with Bad Witnesses: Good and Bad Service



(a) Service starts good, then damps



(b) Service starts bad, then upgrades

Figure A.3.: Results of Integration Test using Experiment 3 with Bad Witnesses: Damping and Upgrading Service

B. Affirmation

Hereby I confirm that I wrote this thesis independently and that I have not made use of any other resources or means than those indicated.

Amsterdam, August 14, 2009

Bibliography

R. Arunachalam and N.M. Sadeh. The supply chain trading agent competition. *Electronic Commerce Research and Applications*, 2005.

A.D. Baddeley and G. Hitch. The recency effect: implicit learning with explicit retrieval? *Memory and Cognition*, 1993.

Boris Beizer. *Software Testing Techniques*. Dreamtech, 2003.

S. Buchegger and J.Y. Le Boudec. The effect of rumor spreading in reputation systems for mobile ad-hoc networks. *Proceedings of WiOpt 03: Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks*, 2003.

S. Buchegger and J.Y. Le Boudec. A robust reputation system for mobile ad-hoc networks. *Proceedings of P2PEcon, June*, 2004.

G. Casella and R. Berger. Statistical inference. *Cole, Pacific Grove, Calif*, 1990.

C. Castelfranchi and R. Falcone. Principles of trust for mas: cognitive anatomy, social importance, and quantification. In *Proceedings International Conference on Multi Agent Systems (Cat. No.98EX160)*, pages 72–79. IEEE Comput. Soc, 1998. ISBN 0-8186-8500-X. doi: 10.1109/ICMAS.1998.699034.

N. Chervany and D. McKnight. The meanings of trust. *Minneapolis (USA)*, 1999.

R. Demolombe. *To trust information sources: a proposal for a modal logical framework*, pages 111 – 124. Kluwer Academic Publishers, 2001.

L. Ding, P. Kolari, S. Ganjugunte, T. Finin, and A. Joshi. Modeling and evaluating trust network inference. In *Seventh International Workshop on Trust in Agent Societies*, 2005.

S. Faehnrich and J. Nimis. How social structure improves distributed reputation systems-three hypotheses. In *Third Intl. Workshop on Agents and Peer-to-Peer Computing (AP2PC04)*, 2004.

M. Feldman, C. Papadimitriou, and J. Chuang. Free-riding and whitewashing in peer-to-peer systems. *IEEE Journal on Selected Areas in Communications*, 2006.

K. Fullam, T. Klos, G. Muller, and J. Sabater. A specification of the agent reputation and trust (art) testbed: Experimentation and competition. *Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*, 2005.

D. Gambetta. *Trust: Making and breaking cooperative relations*. Basil Blackwell, New York, 1990a.

D. Gambetta. Can we trust trust? *Trust: Making and Breaking Cooperative Relations*, 1990b.

Han Guangjie, Choi Deokjai, and Lim Wontaek. *A Reliable and Efficient Approach of Establishing Trust for Wireless Sensor Networks*. IEEE, 2007. ISBN 1-4244-1491-1. doi: 10.1109/ICCP.2007.4352157.

C. Hang, Y. Wang, and Munindar P. Singh. An adaptive probabilistic trust model and its evaluation. In *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems-Volume 3*, pages 1485–1488, 2008.

C. Hang, Y. Wang, and M. Singh. Operators for propagating trust and their evaluation in social networks. In *Proceedings of the 8th International Joint Conference on Autonomous Agents and Multiagent Systems*, 2009.

Joseph Harrington Jr. The social selection of flexible and rigid agents. *American Economic Review*, 1998.

N. Höning, T. Kozelek, and M. C. Schut. Beating cheating: Dealing with collusion in the non-iterated prisoners dilemma. In *The 20th Belgian-Netherlands Conference on Artificial Intelligence*, 2008.

B.A. Huberman and F. Wu. The dynamics of reputations. *Computing in Economics and Finance*, 2003.

S. Johansen. *Trust in initial encounters: a motivational, cognitive theory*. PhD thesis, Bergen, 2007.

Catholijn M. Jonker and Jan Treur. *Formal Analysis of Models for the Dynamics of Trust Based on Experiences*, volume 1647 of *Lecture Notes in Computer Science*, chapter 18, pages 221–231. Springer Berlin Heidelberg, Berlin, 1999. doi: 10.1007/3-540-48437-X_18.

A. Jøsang. A subjective metric of authentication. *Lecture notes in computer science*, pages 329–344, 1999.

A. Jøsang and R. Ismail. The beta reputation system. *Proceedings of the 15th Bled Electronic Commerce Conference*, 2002.

A. Jøsang, R. Hayward, and S. Pope. Trust network analysis with subjective logic. *Proceedings of the 29th Australasian Computer Science*, 2006.

A. Jøsang, R. Ismail, and C. Boyd. A survey of trust and reputation systems for online service provision. *Decision Support Systems*, 2007.

Reid Kerr and Robin Cohen. Smart cheaters do prosper: defeating trust and reputation systems. In *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems*, pages 993—-1000, 2009.

Sarah N. Lim Choi Keung and Nathan Griffiths. Using recency and relevance to assess trust and reputation. *Proceedings of AISB 2008 Symposium on Behaviour Regulation*, 2008.

M. Khambatti, P. Dasgupta, and Kyung Dong Ryu. A role-based trust model for peer-to-peer communities and dynamic coalitions. In *Second IEEE International Information Assurance Workshop, 2004. Proceedings.*, pages 141–154. IEEE, 2004. ISBN 0-7695-2117-7. doi: 10.1109/IWIA.2004.1288044.

M. Kosfeld, M. Heinrichs, and PJ. Zak. Oxytocin increases trust in humans. *Nature*, 2005.

Niklas Luhmann. Familiarity, confidence, trust: Problems and alternatives, 2000.

E.M. Maximilien and M.P. Singh. Toward autonomic web services trust and selection. *Proceedings of the 2nd international conference on Service Oriented Computing*, 2004.

M. Milinski. Cooperation through indirect reciprocity: image scoring or standing strategy? *Proceedings of the Royal Society B: Biological Sciences*, 268:2495–2501, 2001.

Barbara A. Misztal. *Trust in Modern Societies*. Polity Press, 1996.

R. Nelson. Physical and social technologies and their evolution. *Économie Appliquée*, 56: 13—-32, 2003.

M. Nowak and K. Sigmund. Evolution of indirect reciprocity. *Nature(London)*, 7063: 1291, 2005.

M. Nowostawski and N. Foukia. Social collaboration, stochastic strategies and information referrals. In *International Conference on Intelligent Agent Technology, 2007*, volume 55, pages 416–419, Fremont, CA, Mai 2007.

L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web, 1998.

R. Putnam. *Bowling alone: The collapse and revival of American community*. Simon & Schuster, 2000.

P. Resnick, R. Zeckhauser, J. Swanson, and K. Lockwood. The value of reputation on ebay: A controlled experiment. *Experimental Economics*, 9:79–101, 2006.

M.C. Schut. On model design for simulation of collective intelligence. *Information Sciences*, 2009.

K. Sigmund and M. Nowak. Evolution of indirect reciprocity by image scoring. *Nature*, 1998.

Munindar P. Singh and Pinar Yolum. Emergent properties of referral systems. In *2nd International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 592—-597, 2003.

Y. Wang. *Evidence-Based Trust in Distributed Agent Systems*. PhD thesis, Raleigh, 2009.

Y. Wang and Munindar P. Singh. Formal trust model for multiagent systems. In *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI07)*, pages 1551–1556, 2007.

Yonghong Wang and Munindar P. Singh. Trust representation and aggregation in a distributed agent system. *Science*, 2006.

D. Watts and S. Strogatz. Collective dynamics of'small-world' networks. *Nature(London)*, 393:440–442, 1998.

B. Yu, M. Singh, and K. Sycara. Developing trust in large-scale peer-to-peer systems. In *2004 IEEE First Symposium on Multi-Agent Security and Survivability*, pages 1–10, 2004.

Bin Yu and Munindar P. Singh. Search in referral networks. In *Proceedings of AAMAS Workshop on Regulated Agent-Based Social Systems*, 2002.

List of Figures

List of Tables

List of Algorithms