Estimating Reliability of Conditional Promises

Alva L. Couch, Hengky Susanto, and Marc Chiarini

Tufts University, Medford, Massachusetts, USA
alva.couch@cs.tufts.edu, hsusan0a@cs.tufts.edu, mchiar01@cs.tufts.edu

Abstract. Using conditional promises, the reliability of promises can be measured without reference to the reliability of the issuing agent, by defining notions of when conditions operate and when promises are expected to be fulfilled. This inspires an analytical method that attributes promise failures to incomplete knowledge rather than agent unreliability. This analysis allows agents to choose between conditional offers of service based upon statistical measures of the completeness of stated conditions.

1 Introduction

Conditional promises were introduced in [1] to describe situations in which one agent promises to serve another agent whenever its own requirements are met. However, there has been little discussion of how an agent might interpret such conditional promises, or the limits of conditioning as a mechanism. In this paper, we explore how agents can interpret conditional promises and make decisions based upon statistical measures.

The most important claim in the following treatment is that *the reliability of a conditional promise is a more appropriate basis for trust than the reliability of an agent*. Previous work on promises measures "agent reliability" as a statistical function[2–4]. We contend that this is not realistic, because the reliability of promises varies with the kind of promise.

For example, the authors are not reliable concert violinists, and a promise to personally perform a violin concerto is not credible when made by any one of us. However, the authors are reasonably adept at managing systems, so that a promise to manage systems has more credibility. Statistical measures of agent reliability might prohibit us from managing systems, just because we suffer from the delusion that we are also adequate concert violinists!

Our second claim is that *reliability of a specific conditional promise is a measure of completeness of its conditions as a model of its requirements*. This Bayesian world-view is partly inspired by the thinking of E. T. Jaynes[5], and is actually a form of "maximum entropy assumption." Even when probabilistic analysis indicates that a promise is *an unreliable hypothesis*, this is not sufficient evidence that its *source* is unreliable; the promise may be unreliable for external reasons having nothing to do with the originating agent. In Jaynes' methodology, it is an analytical error to use promise reliability as a measure of reliability of the issuing agent without compelling evidence that the agent has complete responsibility for and control over that behavior.

For example, even if we are superb concert violinists, if we do not manage to play the violin concerto we promised to play, this is not incontrovertible evidence of our reliability or lack of reliability as violinists; there may be some other factor (e.g., entropy) contributing to our lack of performance. It could be, e.g., that we have no reliable access to a violin at the time we are expected to play, or even that we have the flu and cannot play because we are ill. It could even be that someone smashed our only violin just before the concert. Thus our promise should not be "I will play a violin concerto", but rather something like "I will play a violin concerto, given that I can obtain a violin, am not ill, it is functional, and I have practiced enough.". The unconditional promise "I will play a violin concerto" is not convincing evidence that the agent is "dishonest" or "unreliable"; it is simply "incomplete"[1].

2 Background

The background and notation for this paper have been developed in [6] (to appear in the same publication and conference) and are briefly summarized here before turning to new material.

A *promise*[1, 4, 7, 8] is a commitment from one sender to one recipient involving one information packet called a "promise body". We can think of each promise as a triple $\langle s, r, b\rangle$ where s is a sender, r is a recipient, and b is a "promise body" describing some commitment of s to r.[2]

A *conditional promise* is a construction in which one sender promises a particular promise body conditionally, based upon the existence of other commitments[6, 4]. In general, we write $(p|q_1, \ldots, q_k)$ for a conditional promise, where p and $q_1, \ldots, q_k$ are "primitive" promises of the form $\langle s, r, b\rangle$.

A particular promise is called *operative* in the context of a particular agent if it is known to be valid in that context, and *inoperative* otherwise. There are two ways to become valid/operative: through being explicitly promised to the agent, or through being a conditional result of other promises becoming operative. All unconditional promises are always operative. The conditional promise $(p|q_1, \ldots, q_k)$ means that p is operative whenever all of $q_1, \ldots, q_k$ are operative.

The typing system for promise bodies b has been studied in [8]. In this paper, we are actually describing part of a theory of types; it would be just as valid to write $(\langle s, r, b\rangle|p)$ as $(\langle s, r, (b|p)\rangle)$; one can (and probably should) characterize the conditions as being part of the body rather than being separate entities.

[1] In our specific case, the promise should be more of the form, "We will play you a violin concerto when pigs fly." There is strong experimental evidence that this is a highly reliable promise, and this does not preclude making some conditional promise later (after one of us has actually learned to play the violin) that has a weaker set of conditions. By issuing promises with inherently false conditions, we might be able to achieve fairly high reliability ratings as agents, but those ratings would be meaningless.

[2] We depart from the pictorial notation used in other papers and utilize traditional graph notation in order to more easily specify derived graphs.

Our notation, however, allows us to think of promises as independent from the body of a promise, which helps us notate complex relationships clearly without defining the type of b.

Each agent's view of the world can be partly represented as a set of conditional and unconditional promises C received by and issued by the agent. One way to think of C is to consider it as a *set of hypotheses* that can be true or false in specific instances.

Definition 1. *For a set of (conditional and unconditional) promises C, we notate the operative promises that result from that set as $\rho(C)$.*

These are the union of the primitive promises in C, together with the consequents p of conditional promises $(p|q_1, \ldots, q_k) \in C$ where all q_i are operative in C. It is also important for us to understand when two representations C and D of promises represent the same situation:

Definition 2. *Two sets of promises C and D are* equivalent *iff $\rho(C) = \rho(D)$, i.e., they represent the same sets of operative promises.*

3 Observability

In the unconditional model of promises, there is no need to consider whether an agent can determine if a promise is operative; all promises are primitive and thus operative when received. In the conditional model, it becomes important to distinguish between what each agent "knows" about the promises that it sends or receives, because conditions that cannot be observed cannot possibly be verified. This work is based upon the concept of observability as first described in [9].

Definition 3. *In the context of an agent X, a promise p is* observable *if X can determine with certainty when p is operative, and* unobservable *otherwise.*

This is a different kind of observability than observing whether a promise is *reliable*; in this case, it is the operative nature only that is observed.

Many promises that can occur as conditions are not guaranteed to be observable. The promise $\langle s_1, r_1, b_1\rangle|\langle s_2, r_2, b_2\rangle$ contains a condition $\langle s_2, r_2, b_2\rangle$ that is not guaranteed to be observable by r_1 unless s_2 or r_2 is equal to r_1. It is not guaranteed to be observable by s_1 unless s_2 or r_2 is equal to s_1. It is not guaranteed to be *mutually observable* by s_1 and r_1 unless either $s_2 = s_1$ and $r_2 = r_1$, or $s_2 = r_1$ and $r_2 = s_1$. Observability of a promise is of little value unless both sender and receiver of each promise can observe equally.

In practice, third-party constructions are common, e.g., "I will give you DNS if I get file service from a third party." In a third-party condition, there is no commitment on the part of the *receiver* to track (or even to have the ability to track) whether the antecedent promises are operative. Consequently, the sender has no knowledge of the receiver's abilities. Thus any promise involving a third-party commitment requires more machinery in order to become operative.

In our previous paper[6], for a primitive promise p, the promise $\langle s, r, \kappa(p)\rangle$ means that s agrees to inform r as to whether p is operative. The promise

$\langle r, s, U(\kappa(p))\rangle$ is a promise by r to use the information about p provided by s. One way to resolve the observability quandary is for each agent to send a $U(\kappa(p))$ promise to each agent that is a potential issuer of a currently unobservable promise p. This is a request for knowledge of whether p has been issued. If each potential issuer responds with a $\kappa(p)$ promise, then observability is assured.

4 Assumptions

In this paper, we assume that:

1. Agents are connected by reliable network connections and all promises are reliably delivered from sender to receiver.
2. All promises in agent conditions are observable (by means of κ or some other abstraction).
3. All operative promises for an agent may be tested via repeated, discrete trials that indicate success or failure.
4. Conditional probabilities of promise success during these trials are stationary (i.e., the probability of success for a particular conditional promise does not vary in time).

These assumptions simplify the following discussion, but leave many issues to be addressed in future work.

5 Reliability

Observability simply means that we can determine whether a commitment is present; whether the commitment is honored is a separate thing.

Definition 4. *From the point of view of an agent X, a promise $\langle s, r, b\rangle$ is* reliable *if whenever it is operative, it is also functional according to some test of behavior (that corresponds to the contents of the promise body b).*

Practical promises cannot ever be entirely reliable, and an agent cannot predict and/or condition its promises against all possible forms of future failure. Many failure modes are not easily detectable themselves, even from the point of view of the promiser.

For example, it is of little value for an agent to try to report that its outgoing communication channels are overloaded, because a message that might *notify* an agent that the problem exists would have to contend with the *cause* of the problem. It is equally amusing to consider how an agent would inform another that an entity *between* them is interfering with or even spoofing their attempts at communication. But in practical terms, "interference" can take much more subtle forms, including presence of unknown bugs in software, latent conditions in configuration exposed by client-server interactions, etc.

Computing the experimental reliability of an unconditional promise is straightforward.

Definition 5. *The reliability of an unconditional promise is the probability that a trial of the promise commitment will succeed.*

If reliability is stationary (i.e., the probability is not time-varying), one estimate of reliability is the ratio of "number of successes" to the "number of trials". This becomes an estimate of "average reliability" rather than "reliability" whenever reliability varies with time.

To compute the reliability of a conditional promise, one must account for whether conditions are operative:

Definition 6. *The* reliability *of a conditional promise is the probability that its commitment will be delivered* when it is operative *in the context of the observing agent.*

The experimental reliability of a conditional promises is thus the ratio of "number of successes while conditions are operative" to "number of trials while conditions are operative".

It is important to distinguish between a "failed promise" whose trials indicate failure rather than success, and a "broken promise" that is contradicted by some newer promise. Whether a promise succeeds or fails contributes to its reliability, but if it is explicitly broken, no more measurement of its success is meaningful because the commitment it represents is no longer present. Reliability measurements must account for changes in which promises are operative due to broken promises and other forces, such as time-varying promise scoping[6].

In other work, the reliability of a promise is combined with that of other promises to obtain a concept of agent reliability. While this is a sound idea when considering security of a system, it does not represent the case where an agent simply does not know enough to appropriately condition a promise. It is safe to assume that an agent *cannot* know the complete conditions under which a promise will be fulfilled, so that the failure to fulfill a promise is a *failure of agent knowledge* rather than *failure of an agent.*

One can embody this assumption by thinking of a failed promise p as having some invisible condition, call it $\varepsilon(p)$, that causes the failure by its absence. If S is a set of conditions that we think should be operative before p is operative, we can think of the "realistic" promise $p|S$ as having imperfect reliability, while the "ideal" promise $p|S \cup \{\varepsilon(p)\}$ exhibits perfect reliability[3]. To understand how unreliable a specific promise can be, we can quantify the probability that S is operative while ε is not. A failure of p with S operative means that $\varepsilon(p)$ is *not* operative for some undisclosed reason[4]. It is also possible that $\varepsilon(p)$ represents time-varying or unobservable conditions.

This world-view has a profound effect upon how we analyze sets of conditional promises. As a human example, when we promise to pay our tax bills, we do not mention the idea that we might not have enough money at some point in the future. This "sin of omission" might be considered the true source of unreliability, *not the agent.* This omission is part of the content of ε.

[3] The resemblance between $\varepsilon(p)$ and system entropy is purely intentional!

[4] It is perhaps useful to consider $\varepsilon(p)$ as the "set of latent preconditions of p"[10].

Another example is that of a web server W that promises service to client M with the constraint of "delivery $<$ 100ms" and the condition "fileserver F delivery $<$ 25ms". If F's promise is inoperative (and observable by W), then so is that of W, and M is aware that it has no guarantee. However, it may be the case that W constrains itself to less than 50 simultaneous client requests at any one time *without* telling M. This is essentially a promise to itself. Suppose that M obtains the promise of service from W and attempts to use that promise as the 51st client. Service will be denied and the promise will fail, because the *hidden* ε condition "number of concurrent requests $<$ 50" is inoperative.

6 Condition Graphs

To compute sample probabilities of conditional promise reliability, one can use condition graphs. A condition graph is a representation of the dependencies between promises held or promised by a specific agent.

Definition 7. *The "condition graph" $G = \langle P, Q \rangle$ corresponding to a set of conditional promises C is formed as follows. The nodes of the condition graph are subsets of primitive promises; P is a subset of the powerset of all primitive promises contained in C as unconditional promises or primitive elements of conditionals. For each conditional promise $p|S$ in the graph, where S is a set of promises that must become operative before p is operative, construct a directed edge in G from S to $\{p\}$ (the singleton set containing p).*

In other words, there is an edge in G for each *way* in which a promise p can become operative. If a node p in G has two incoming edges, this represents two ways in which the node can become operative, e.g., $p|q$ and $p|r$. The edge $(\{q, r\}, \{p\})$, by contrast, indicates that both q and r must be operative before p is operative under that rule. All arrows are sufficient but not necessary, and p can become operative by other means, including simply being promised unconditionally in C. Figure 1 shows a simple condition graph. We indicate unconditional promises via boldface circles around symbols.

Condition graphs allow one to calculate dependencies between promises.

Definition 8. *In a condition graph G, a subset of promises S_1* controls *a subset of promises S_2 if whenever every promise in S_1 is operative, every promise in S_2 is also operative.*

As an example, consider the graph for $\{(p|q, r), (q), (r|s)\}$. In this graph, $\{s\}$ controls $\{p\}$, because if s is operative, r is operative, and q is always operative, so p is operative. But we could also say that $\{s\}$ controls $\{p, q, r\}$. The notion of control is that of a guarantee; if s is operative, what other promises are guaranteed to be operative as well?

One construction will greatly reduce the complexity of calculating conditional relationships.

Definition 9. *Two condition graphs G and H are* equivalent *iff they express the same control relationships.*

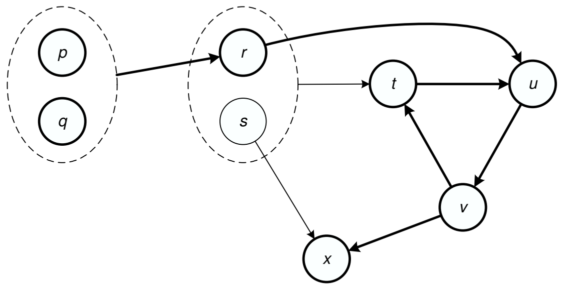

Fig. 1. A simple condition graph comprising the set of promises $\{(r|p,q),(t|r,s),(t|v),(u|t),(u|r),(v|u),(x|v),(x|s)\}$. p and q are unconditional promises and are thus operative. Following the operative edges, we see that every promise except s is made operative. s is inoperative because it is not promised unconditionally.

Lemma 1. *Any condition graph G with a strongly connected component V of primitive promises is equivalent with a graph G' in which V is represented as a single node (set) v.*

Proof. Recall that a strongly-connected component of G has the property that there is a path from any node to any other. In a condition graph, these paths represent control relationships. Thus if any one of these nodes becomes operative, all others are operative as well. We construct G' from G as follows:

1. Add a new node v to the graph, representing the set of nodes in the strongly-connected component.
2. Change all edges pointing to nodes in V (including those within the component) to point to v.
3. Remove V.

Claim: this new graph embodies the same control relationships. To see this, consider what happens when any one node $n \in V$ becomes operative. In G, this makes the component operative. In G', this has the same effect, in the sense that the set representing the component becomes operative, because of the arrow from n to v. Any other arrow into a member of V in G has the same effect when pointed at v in G'. □

It is sometimes useful to connect the graph with more edges than the explicit promises allow. The most important implicit relationship is that of subsets.

Definition 10. *The* completion $\omega(G)$ *of the condition graph G includes the edges in the condition graph, as well as edges from set S_1 to set S_2 whenever $S_1 \supset S_2$.*

It should be obvious that this does not change inferences of what is operative: $\rho(\omega(G)) = \rho(G)$.

Completions also make it easy to describe how to compute control dependencies:

Theorem 1. *In a condition graph G, the maximal set of nodes controlled by a subset S is the union of every subset S' reachable from S in the completion $\omega(G \cup \{S\})$.*

Proof. The point of the completion is to take every subset relationship into account. We can demonstrate this by induction on the size of the control subset S. If $|S| = 1$, the lemma is trivially true, because all promises that S controls are directly connected by paths of singletons (recalling that all edges in the completion lead to smaller subsets and singletons). If the lemma is true for $|S| \leq k$, then for $|S| = k + 1$, we can compute the sets of controlled promises by looking at the edges exiting S and from all subsets of S that appear within the completion. Since these edges always connect to sets of size less than S (by construction) the inductive hypothesis applies to these sets, and the union of all operative nodes found by following the paths from each of these sets is maximal.

The point of this lemma is to give an easy way to compute the effect of making a set of nodes operative. To compute the effects, one utilizes breadth-first search of the completion graph, progressing along all paths starting at the set and stopping when cycles are detected.

7 Computing Reliability

In simplifying the calculation of reliability, we are aided by condition graphs. The edges in the graph are control relationships, so each edge can be labeled with a tally of successes and failures, as well as a binary flag denoting whether the source of the edge is operative. Each time a promise is tested for success or failure, we increment the tallies on all *incoming* edges that are operative. This is a quick operation once the condition graph is built, even if promises become inoperative over time, as described in [6].

Observations made by one agent are less useful than observations made over a population of agents. For promise sets consisting of only primitive promises, it is valuable to average the accuracies for promises $\langle s, r, b\rangle$ as measured by each recipient r, as a measure of the reliability of agent s [2]. For conditional promises, we must consider instead how one would cluster tuples of the form $\langle s, r, b\rangle|S$. One way to cluster these is to hold s, b, and S constant and tally over all r, as a measure of the reliability of promises of the kind $\langle s, *, b\rangle|S$.

In clustering statistics for several agents, it helps to have a notion of condition graph that spans agents.

Definition 11. *Two promises p and q held by different agents are* comparable *if one can transform p into q by replacing each reference to the recipient of p with a reference to the recipient of q.*

For example, $p = \langle s, r, b\rangle|\langle r, s, c\rangle$ and $q = \langle s, t, b\rangle|\langle t, s, c\rangle$ are comparable, because one can transform p into q by replacing r with t.

Each set of comparable promises forms a *promise pattern* originating from a specific sender and containing a free variable for the recipient. In the case above, if the free variable is X, the pattern is $\langle s, X, b\rangle|\langle X, s, c\rangle$.

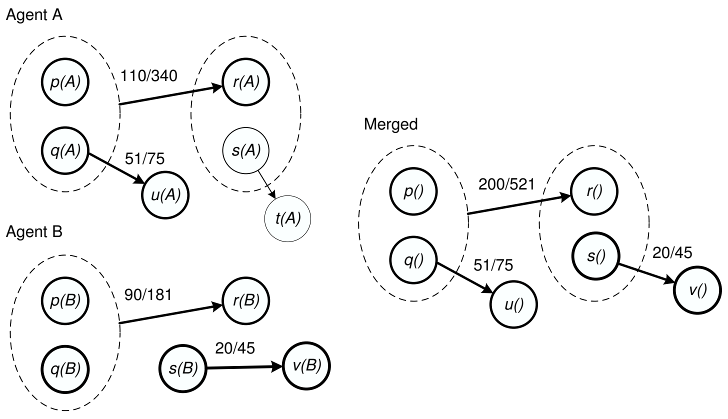

Fig. 2. Clustering statistics from multiple agents requires adding tallies for edges representing comparable promises.

Definition 12. *Given a set of condition graphs from multiple agents, the* merged condition graph *is formed as follows (Figure 2):*

1. *Merge comparable nodes into single nodes.*
2. *In each case where all nodes in a set are comparable between two or more agents, merge comparable sets.*
3. *Replace edge label statistics (successes/total trials) exiting merged nodes or sets with sums of statistics (over both numerator and denominator) for the merged nodes or sets.*

This accomplishes the same calculation that we described before: the graph represents statistics for particular $\langle s, *, b|S\rangle$ patterns.

In the merged condition graph, the sample probability labeling each edge is a measure of the reliability of a promise pattern originating from a particular server. This can be used to globally compare reliability of the same service from different servers, or to test hypotheses about causes of unreliability.

8 Reasoning about Reliability

Agents can utilize annotated condition graphs to reason about whether promises are appropriately conditioned or not. When the set of promises C contains two different conditional promises with the same consequent p, it is possible to compute which set of conditions is more reliable in assuring p over time.

Figure 3 depicts an example of this calculation. In (a), an agent has observed the success of condition sets $\{s, t, u\}$ and $\{s, t, u, d\}$ over ten trials, where d becomes operative and inoperative during the trials due to broken and resent promises or time-varying constructions[6]. Promise f is not considered, as it remains inoperative throughout all trials. After 1000 trials (b), the success rate indicates that when d is operative, the promise tends to fail *more* often. Since the set of promises $\{s, t, u\}$ is sufficient to make p operative, the agent may wish to negotiate that future offers of p not be conditioned upon d, so that d can remain inoperative.

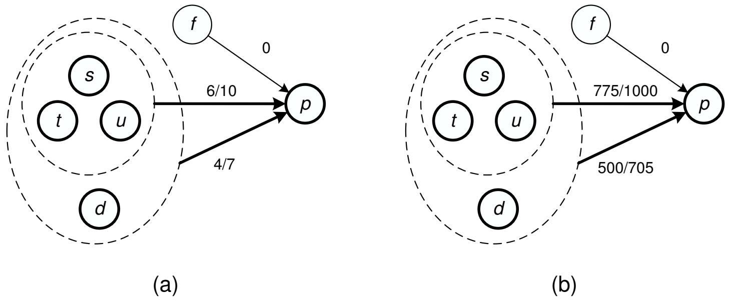

Fig. 3. Reliability estimates for conditional promise p after (a) 10 trials and (b) 1000 trials.

More generally, the question arises as to whether any agent contains conditions that could make a promise more reliable or unreliable than those stated by its sending agent. It is possible for agents to calculate reliability measures of *hypothetical* conditional relationships other than those asserted by the sending agent, and to experimentally determine the "best set of conditions" for assuring a behavior.

Suppose, for example, that we suspect that a particular promise p is a hidden prerequisite for promise $q|S$, even though it does not appear in S as a prerequisite. We construct a promise pattern for the promise $q|S$ and locate all agents holding comparable promise. For each comparable promise $q'|S'$, we instruct the agent to form an extra edge representing the *hypothetical* promise $q'|S' \cup \{p\}$. This hypothetical promise is constructed so that instances on every agent are comparable, so statistics can be gathered and combined to form a global estimate of its reliability.

9 Conclusions

We have presented a new method for analyzing the reliability of promises issued by an agent, to replace methods for analyzing agent reliability as a whole. Reliability is analyzed relative to how an agent *scopes* and *qualifies* its offers for service, which gives agents the opportunity and motivation to learn how to qualify their behaviors more accurately. If agents are responsible only for their actions, then we can distinguish between "good" and "bad" agents, but if they are instead responsible for their knowledge, they can grow in knowledge over time, and make more intelligent promises as a result.

In this paper we have assumed that there is perfect observability of promises by each agent, reliable communications between all agents, and that arbitrary promises can be observed and tested. Without these assumptions, one cannot always easily combine reliability results from different agents. For example, suppose we have two agents X and Y holding promises $p|S$ and $p|S'$, respectively. The reliability of $p|S$ and $p|S'$ are likely to differ for unequal S and S'. But even if X and Y hold the same promise $p|S$, it is possible that their abilities to observe S differ for some reason external to the promise $p|S$. E.g., one agent might not hold the necessary "knowledge"(κ) promises, or unreliable network communications might make κ commitments impossible to honor, leading to non-comparable measures of reliability. In order for us to cluster and combine statistics collected by multiple agents, the promise, the observability of its conditions, and the ability to observe failures must be the same for both agents. This is a matter for future study.

As well, real systems do not obey stationary probability distributions and exhibit entropic behavior as a result of causes other than holding or not holding promises. The "entropy condition" $\varepsilon(p)$ to which we have neatly attributed all failures of conditional promises does not consist solely of promise conditions, and we cannot eliminate that condition through promises alone. It is not yet clear how to determine whether our model of promise-based conditions is as complete as possible or not.

What is the limit of this technique? Perfect reliability is impossible, and near-perfect reliability may well require a large number of conditions. But agents can do what humans do already: promise based upon conditions that seem to be most important. Principal axis analysis could easily identify these conditions.

By considering the reliability of promises, rather than reliability of the agent, we allow agent knowledge to evolve over time, and allow future networks to accept value wherever it evolves.

References

1. Burgess, M.: An approach to understanding policy based on autonomy and voluntary cooperation. In Schönwälder, J., Serrat, J., eds.: DSOM. Volume 3775 of Lecture Notes in Computer Science., Springer (2005) 97–108
2. Bergstra, J., Burgess, M.: Local and global trust based on the concept of promises. Technical Report PRG0606, University of Amsterdam (September 2006)
3. Burgess, M., Fagernes, S.: Pervasive computer management: A smart mall scenario using promise theory. In: Proceedings of First IEEE International Workshop on Modelling Autonomic Communication Environments (MACE2006). (2006) 133–
4. Burgess, M., Fagernes, S.: Pervasive computer management: A model of network policy with local autonomy. IEEE Transactions on Networking (2006) (submitted)
5. Jaynes, E.T.: Probability Theory: the Logic of Science. Cambridge University Press (2003)
6. Couch, A., Susanto, H., Chiarini, M.: Modeling change without breaking promises. To appear in Proceedings of AIMS-2007 (February 2007) (preprint)
7. Burgess, M., Begnum, K.: Voluntary cooperation in pervasive computing services. In: LISA, USENIX (2005) 143–154

8. Burgess, M., Fagernes, S.: Promise theory - a model of autonomous objects for pervasive computing and swarms. In: ICNS, IEEE Computer Society (2006) 118
9. Couch, A., Sun, Y.: On observed reproducibility in network configuration management. Science of Computer Programming **53** (2004) 215–253
10. Kanies, L.: Isconf: Theory, practice, and beyond. Proceedings of the Seventeenth Systems Administration Conference (LISA XVII) (USENIX Association: Berkeley, CA) (2003) 115