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 considering the reliability of the whole agent, by defining notions of when conditions operate and when promises are expected to be fulfilled. This inspires an analytical method based upon "incomplete conditions" that attributes failures to incomplete knowledge rather than operational faults. This analysis allows agents to choose offers 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 if 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. We define a concept of observability and knowledge propagation, and a notion of exactly when a conditional promise is binding on a host. This leads to a concept of reliability for a promise that is to some extent independent of the promising agent. The reliability of a promise, considered as a transaction originating at one sender with one kind of commitment, can be analyzed *without* reference to the reliability of the sender as a whole.

The most important claim in the following treatment is that *the reliability of a promise is a more appropriate measure of reliability 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. The authors are not reliable concert violinists, and the promise to perform a violin concerto is not credible when made by any of us. However, the authors are reasonably adept at managing systems, so that a promise to manage systems has more credibility. Current statistical measures of agent reliability would prohibit us from managing systems, just because we might believe incorrectly that we are adequate concert violinists!

2 Background

A *promise* is a transaction between one sender and one recipient involving one information packet called a "promise body". Thus 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 body describing

some commitment of s to r[1]. The typing system for promise bodies has been extensively studied in [5].

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

The consequent of a conditional promise is *operative* if the conditions (antecedents) are *known* in context to be operative themselves, and *inoperative* otherwise. An operative promise is one that is actually usable, rather than hypothetical[2]. All unconditional promises are operative; they can be considered as conditionals with no listed conditions.

Each set of conditions is a conjunction; one could write $p|q_1, \ldots, q_k$ as $p|q_1 \wedge \ldots \wedge q_k$. Disjunctions are indicated by sets of conditionals, e.g., $\{p|q, p|r\}$ represents that p is operative if either q or r is operative, or perhaps both $(p|q \vee r)$. Exclusive-or is handled by using a negation operator $\neg$, where $\neg X$ is operative exactly when X is not. The exclusive-or construction $p|q \oplus r$ is represented as a set of two promises: $\{(p|q, \neg r), (p|\neg q, r)\}$. A promise whose consequent is a conjunction is representable as a set of two promises: $p \wedge q|r$ is equivalent with $\{p|r, q|r\}$. In this way, we see that most common logical conditions can be represented in a canonical form.

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

It is also important for us to understand when two representations of promises "mean" the same thing:

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

The observability of a promise is relative to each agent. The set of promises that an agent considers operative need not agree with that of another agent. Although a sender may make a conditional promise in good faith, the receiver can only act upon that promise if *it* can observe whether conditions are operative or not. A conditional promise for which the receiver cannot observe all prerequisite conditions can never become operative from that agent's point of view[3], though the consequent may become operative via some other promise.

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

[2] In [4], the notation $T(p)$ is used to indicate that "p is true", as a conditional promise. We adopt a simpler language to more carefully distinguish between being operative and inoperative, independent of whether a promise is conditional. In particular, p is not operative for an agent X if X has no knowledge of p.

[3] E.g., "I will pay you when pigs fly."

3 Observability

In the unconditional model of promises, there is no need to consider whether an agent can determine if a promise is operative. 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.

It is thus important to know with what certainty one can observe whether a promise is operative.

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.*

In general, 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 *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 $r_2 = s_1$. A sender and receiver can observe promises between themselves, but not between others.

Note that observability of a promise simply means that one can tell with certainty whether it is operative, but not whether the commitment that it represents is fulfilled. This is of little value unless both sides of a transaction can observe equally.

Definition 4. *A primitive promise $\langle s, r, b\rangle$ is* mutually observable *if it is observable on both s and r.*

Trivially, any primitive promise involving *both* s,r is mutually observable by s and r.

Certain kinds of promises are inherently more observable than others. In particular, a promise that contains only conditionals exchanged between sender and receiver is inherently more observable than one conditioned upon the behavior of possibly unobservable third parties. If conditionals represent total knowledge, then any such promise is fully observable. The case in which promises involve third parties is more complex, and best described through a convenient assumption:

Postulate 1. *A conditional promise $p|q_1, \ldots, q_k$, where $p = \langle s, r, b\rangle$, includes two commitments:*

1. *A commitment to provide the promise b under the prerequisite conditions.*
2. *A commitment to maintain knowledge of the prerequisite conditions.*

This postulate is trivially satisfied if $q_1, \ldots, q_k$ are all promises between s and r, because in that case, both agents can fully observe all conditionals. However, if q_i involves a third party,

1. There is an implicit commitment from the sender s to maintain knowledge of whether q_i is operative.
2. There is a need for the receiver to obtain that knowledge in order for the promise ever to become operative.

Unfortunately, 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 prior work[6], $\kappa(p)$ is a promise body designed to transfer knowledge of whether the promise p is operative, e.g., the promise $\langle s, r, \kappa(p)\rangle$ would mean that s agrees to inform r as to whether p is operative. This promise, in turn, depends upon s's *ability* to determine whether p is operative, which depends upon the promises that s itself holds.

This is a bit different than the notion of a "coordination promise" as described in [4]. $\kappa(p)$ represents a directional commitment from one agent to another to transmit knowledge of whether p is operative, independent of the nature of p's body, its type, or its purport. By contrast, $C(b)$ is an agreement to coordinate values of a particular parameter b in a configuration. In other words, $C(b)$ is an operation in which several agents agree to coordinate values having particular *types* and constraints described by b, rather than knowledge of whether a *promise* $p = \langle s, r, b\rangle$ is operative or not.

One simple way to resolve the observability quandary is for each promising agent s to send a $\kappa(p)$ promise for each third-party promise in its conditions. The sender has the ability to do this via our postulate. Then, if the receiver responds with $U(\kappa(p))$, there is a binding between server and client regarding the state of the third-party promise and the consequent of the condition can become operative.

4 Reliability

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

Definition 5. *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 exterior model of behavior (specified through the contents of the promise body b).*

Practical promises cannot ever be reliable. An agent cannot predict and/or condition its promises against all possible forms of future failure. Many of these 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 the *notification* of the state 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.

In prior work, reliability is considered to be a property of an *agent*. In our work, we consider it a property of each *conditional promise*. Without loss of generality we can consider each such promise a construction $p|S$ where p is a primitive promise and S is a set of primitive promises. An unconditional promise is just a conditional promise with an empty set of conditions S.

Definition 6. *The reliability of an unconditional promise is the probability that its commitment will be delivered, over time or trials, as appropriate.*

This just means that one can accumulate statistics on a particular promise, as before. But now a tricky manipulation is necessary:

Definition 7. *The* reliability *of a conditional promise is the probability that its commitment will be delivered* when it is operative *according to the listed conditions.*

In previous theory, the reliability of a promise was combined with that of others to obtain a concept of agent reliability. While this is a sound idea when considering security of a system, it does not represent the simple case where an agent does not know enough to appropriately condition a promise. We 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 knowledge* rather than *failure of an agent*.

We embody this assumption by thinking of a failed promise as having some invisible condition, call it ε, that causes the failure by its absence. Thus we can think of the "realistic" promise $p|S$ as having imperfect reliability, while the "ideal" promise $p|S \cup \{\varepsilon(p)\}$ represents the risk of it not being fulfilled as a separate knowledge condition[4]. To understand how reliable a specific promise can be, we must 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[5].

This world-view has a profound effect upon how we analyze sets of conditional promises. Any particular promise *will* be broken by some sets of exterior conditions. A promise is reliable to the extent that the set of listed conditions is sufficient. 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 the broken promise, *not the agent*. This omission is part of the content of ε.

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 client requests at any one time. This is essentially a promise to itself. Suppose M obtains the promise of service from W and attempts to use that promise as the 51st client. Service will be denied

[4] The resemblance between ε and system entropy is purely intentional!

[5] It is sometimes useful to consider $\varepsilon(p)$ as the "set of hidden preconditions of p".

and the promise will be broken, but not because of stated conditions becoming inoperative or the death of W; the *hidden* ε condition "number of concurrent requests < 50" is inoperative.

5 Clustering

The key to any statistical analysis is to define how things will be clustered or considered comparable elements of a population. Former methods for statistical analysis[2] considered the primitive promise $\langle s, r, b\rangle$ as the appropriate unit of clustering. In that case, it is obviously valuable to sum over the reliabilities measured by each r.

Here, instead, we are forced to consider how one would cluster tuples of the form $\langle s, r, b\rangle | S, \varepsilon(\langle s, r, b\rangle)$. In contrast to definitions in [7], the shared value of a conditional promise can be based upon how well it guards against potential failures and/or advertises them via conditions. Thus statistics for $p|S$ and $p|S'$ are likely to differ for different S and S'.

Thus an appropriate unit of clustering is to consider pairs, consisting of a sender and a conditional promise, over all receivers and/or all instances of the promise acted upon by one receiver. The reliability of a particular conditional promise, received from a particular host, is the sample probability that the promise will be honored based upon direct observation and the observations of peers.

But how can this reasonably be calculated, given that conditionals can depend upon one another? It would seem a complex calculation to compute any kind of sample probability. The key is to exploit control relationships between promises within an agent.

6 Condition Graphs

A condition graph is a representation of the dependencies between promises held or promised by a specific agent. In this section, we develop the basic properties of these graphs, which suggest a simple way to calculate the effects of dependencies between promises for an agent.

Definition 8. *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 in C. 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 $\{p\}$.*

In other words, there is an edge in G for each *way* in which a promise 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 $p|r$, as discussed above. The edge $(\{q, r\}, \{p\})$, by contrast, indicates that both q and r must be operative

before p is operative under that rule. This expresses that $q \wedge r \rightarrow p$. 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.

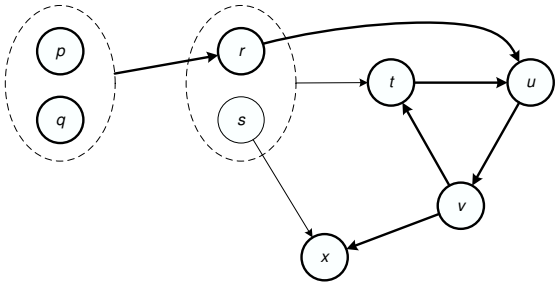

Fig. 1. A simple condition graph comprising the set of promises $\{(r|p,q),(t|r,s),(t|v),(u|t),(u|r),(v|u),(x|v),(x|s)\}$. p and q are unconditional promises and are thus operative. Following the operative edges, we see that every promise except s is made operative. s is inoperative due to other conditions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

Condition graphs allow one to calculate dependencies between promises.

Definition 9. *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 10. *Two condition graphs G and H are* equivalent *iff they express the same control relationships.*

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' 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.

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'. □

This allows us to assume a useful property:

Theorem 1. *Any condition graph G is equivalent with a graph G' that is acyclic.*

Proof. Cycles are strongly connected components. Utilize the construction of Lemma 1 to remove each cycle in turn. □

So without loss of generality, we can assume that a condition graph is an acyclic relation between subsets of primitive promises.

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 11. *The* completion $\chi(G)$ *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: $\rho(\chi(G)) = \rho(G)$. But it does make the graph properties easier to express.

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

Theorem 2. *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 $\chi(G \cup \{S\})$.*

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.

Note that in the definition above, a promise that is not conditional is implicitly "controlled" by all promises. This is counter-intuitive, but removes some cases from proofs.

7 Computing Reliability

In simplifying the calculation of reliability, we are aided by condition graphs. The reliability of a promise is always modal and changes with the conditions and restrictions that one places upon it. Thus an appropriate tally for a conditional promise is to count the ratio of the number of times it is fulfilled to the number of times it is tried, over only the trials for which the conditionals indicate that the promise is operative.

A simple way to perform this calculation is to utilize the condition graph. We note that 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, 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 modeled in [6].

Figure 2 shows reliability estimates for a promise p. In (a), an agent has observed the success of condition sets $\{s, t, u\}$ and $\{s, t, u, d\}$ over ten trials. Promise f is not considered as it remains inoperative throughout all trials. After 1000 trials (b), the success rate indicates that d is a contra-variant condition. 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 on d. Note that the trials need not be conducted by the agent itself. It may choose to query all peers that receive similarly conditioned classes of promises. The mechanisms governing these exchanges are left for future work.

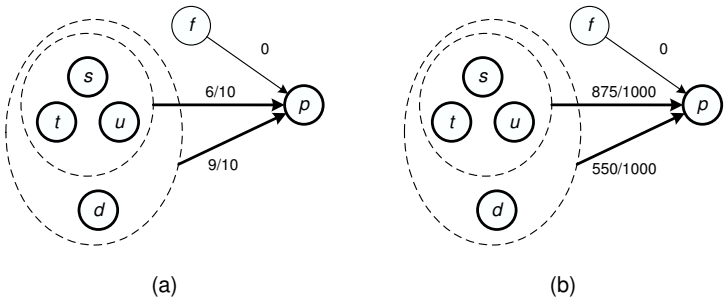

Fig. 2. Reliability estimates for conditional promise p after (a) 10 trials and (b) 1000 trials.

8 Conclusions

The old way of thinking about promises is that the promise itself is a thing to be analyzed, and the agent a thing to be assessed. The basic principle of the theory is that agents should not be disassembled in order to understand the function of their interactions.

We propose a new method for analysis that – without delving into the internals of an agent – renders them culpable for how they *frame* and *scope* their offers for service. This is completely within the spirit of the original theory, but allows one to analyze offers for different services as decoupled and distinct actions of one agent, deserving of separate analysis.

If agents are culpable for their actions, then we can distinguish between "good" and "bad" agents, but if they are culpable for their knowledge, they can grow in knowledge over time, and make more intelligent promises as a result.

For example, suppose an agent has promised something that has turned out to be a bad bet. In the old theory, the agent would have to die and be reborn in order to reset its statistics and make good promises again. In our framework, the agent could change what it is promising to incorporate the new conditions as it learns them, hence learning about its environment and incorporating this new knowledge into its offers.

What is the limit of this? Perfect reliability is impossible, and describing near-perfect reliability nearly so. 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.

By considering the promise, rather than the agent, we allow future networks to accept value wherever it occurs.

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 (MACE). (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. Burgess, M., Fagernes, S.: Promise theory - a model of autonomous objects for pervasive computing and swarms. In: ICNS, IEEE Computer Society (2006) 118
6. Couch, A., Susanto, H., Chiarini, M.: Modeling change without breaking promises. In: Proceedings of AIMS. (February 2007) (submitted)
7. Burgess, M., Fagernes, S.: Voluntary economic cooperation in policy based management. IEEE Transactions on Networking (2006) (submitted)