2016年度 首藤研究室 輪講 第8回

高橋　良希

2016年 5月 9日

概 要

P2P システムに**信頼管理システム**を導入することで，前節で述べたような問題を避けることができる．この章では，**認証情報**に基づいた信頼管理システムとして，*PolicyMaker* や *Trust-*$\mathcal{X}$ を紹介し，**評判**を用いた信頼管理システムとして，*P2PRep* や，*XRep*，*NICE*，*PeerTrust*，*Regret*，*NodeRanking*，*XenoTrust*，*EigenRep*，*P-Grid* を紹介する．

7.1　概念

信頼 (*trust*) とは，「人，組織，ものの，誠実さや，良し悪し，能力，安全性に対する信条や自信である」と定義できるが，これはユニークでなく異なる視点ににによって異なる定義が用いられる．

7.2　信頼モデル

一般的に，信頼モデルは**認証情報** (*credential*) に基づいた信頼と，**評判** (*reputation*) に基づいた信頼に分類することが可能である．

7.2.1　認証情報に基づいた信頼モデル

エージェント (*agent*) が他のエージェントを信頼するべきか否かを判断するとき，そのエージェントの認証情報を確認する．**方針** (*policy*) を満たす場合はそのエージェントの行動は信頼でき，そうでない場合はそのエージェントは使用するべきでない．コンピュータシステムで最も用いられる認証情報システムは，**秘密・公開鍵システム** (*public/private keys system*) である．このシステムでは，まずエージェントがシステムに参加する際に公開鍵と秘密鍵を生成し，その後，公開鍵はエージェントの情報と共に信頼できる**仲間** (*party*) に保存され，秘密鍵はエージェントの識別子として秘密に保持される．エージェントが仕事を行うときは，そのエージェントは秘密鍵で情報を記述することで，相手の認証情報を確かめることが可能になる．

7.2.2　評判に基づいた信頼モデル

多くの場合，信頼を確かめるためには，その人の過去の行動を考慮する必要がある．評判の観念は広く社会に用いられ，各参加者は**評判値** (*reputation score*) を有している．*eBay* や *Amazon Auction* では，ユーザに**フィードバックチャンネル** (*feedback channel*) を提供し，各取引後に売り手と買い手が互いに評価する．結果的に，評判値によって人々はその人を信頼できるか否かを判断することが可能になる．評判と，評判に基づく信頼は次のように定義される．

- 評判とは，エージェント y が，エージェント x の過去の取引を通して見た x に対する知覚である．**局所評判** (*local reputation*) は，x と y の間の取引のみから見た，y の x についての印象である．**大域評判** (*global reputation*) は，x が全システム内のエージェントと行った過去の全ての取引から y が得た評判である．
- 信頼とは，エージェント y が，エージェント x の取引の成功において x を信用することである．もし x の評判が良い場合，y は x を信頼し，x の評判が悪い場合，取引において y は x を信頼するべきでない．

7.3　認証情報に基づいた信頼システム

7.3.1　PolicyMaker

7.3.1.1　システム構造

PolicyMaker では，システム内のエージェントは，**局所方針** (*local policy*) や認証情報を指定することができ，システムはこの方針や認証情報に基づいて，許可される行動に対しては「yes」と返し，許可されない行動に対しては「no」と返す．

7.3.1.2　PolicyMaker Language

PolicyMaker では，局所方針や認証情報を**表明** (*assertion*) として次の構文で記述する．

$$Source\ ASSERTS\ AuthorityStruct\ WHERE\ Filter$$

ここで，*Source* は方針の出処を示し，*AuthorityStruct* は表明を適用する公開鍵を示し，*Filter* は公開鍵に対して *Source* が信頼可能な行動を記述する．エージェントが他のエージェントの行動を確かめたいとき，エージェントは次の形式のクエリを Policy Maker に送信する．

$$key_1, key_2, \ldots, key_n\ REQUESTS\ ActionString$$

ここで，*ActionString* は公開鍵に求められる信頼される行動を記述する．

7.3.1.3 クエリ処理

方針の出処や公開鍵をノードとし，フィルタをエッジとした有向グラフ G に表明を結び付ける．ある表明について，source を s，authority を a，filter を f とすると，これらはノード s, a と，f でラベル付けされた有向エッジ $s \to a$ で表される．キー $k_1, k_2, \ldots, k_n$ と行動 t を含むクエリに対しては，s から，入力が $k_1, k_2, \ldots, k_n$ で，行動 t を含む終点ノード d へのパスを見つける処理を行う．

7.3.2 Trust-$\mathcal{X}$

Trust-$\mathcal{X}$ は，認証情報や方針を指定するために，*$\mathcal{X}$-TNL* と呼ばれる XML ベースの言語を用いるフレームワークを提供する．Trust-$\mathcal{X}$ は，**信頼チケット** (*trust ticket*) とキャッシュを用いることで，信頼の検証の速度を向上させる．

信頼チケット	成功した各取引後に，エージェントによってその相手に与えられる特別な認証情報である．相手は以前の取引に関連するリソースの**交渉処理** (*negotiation process*) を早めることが可能になる．
キャッシュ	複数のエージェントが同じリソースを尋ねると，それらの交渉処理は同じになることがあり，キャッシュは交渉処理の準備の時間を減らすことに役立つ．

7.3.2.1 システム構造

Trust-$\mathcal{X}$ フレームワークでは，各実体は**証明書** (*certificate*) のプロファイルを持ち，各証明書は認証情報か**宣言** (*declaration*) のいずれかとなる．交渉処理は次の 4 つの段階で処理される．

- **導入** (*introductory*): 二つのエージェントの取引を成立するために必要なコンディションを調べる．クライアントエージェントはサーバエージェントの要求するリソースの性質を調べ，サーバエージェントはクライアントエージェントのコンディションを調べる．
- **列の生成** (*sequence generation*): 両者の方針を満たすリソースを取得するための両者の証明書の列が決定される．もしキャッシュに同様の取引がある場合，列はキャッシュから取得され，エージェントが信頼チケットを保有する場合，リソースは即座に受け渡される．
- **証明書の交換** (*certificate exchange*): 列が生成され両者で承認されると，証明書の交換が行われる．一度証明書が調べられ，満たされると，要求されたリソースは受け渡される．
- **信頼列のキャッシュ** (*caching of trust sequence*): この取引の信頼列をキャッシュする．

7.4 個人評判に基づいた信頼システム

7.4.1 P2PRep

7.4.1.1 基本的な投票プロトコル

投票プロトコル (*polling protocol*) は次のフェイズからなる．

- リソース探索: ソースを探すピアはクエリを隣接ノードに送信し，そのピアはさらに他のピアに送信する．クエリを受け取ったノードが探索コンディションを満たすリソースを含む場合，送信元にメッセージを返信する．
- リソース選択と投票: 返信結果からピアのリストを選択し，システム内の他のノードにこれらのピアの評判を尋ねる *Poll* メッセージをブロードキャストする．これを受け取ったピアは，自身の情報から尋ねられたピアの評判を調べ，要求元に *PoolReply* を送信する．
- 票の評価: 悪意あるピアは投票処理に干渉するため，疑わしい票は除去する．さらに，要求元のピアは，投票者のピアに *TrueVote* メッセージを送信し，*TrueVoteReply* メッセージが返信されると，最も評価の良いリソース提供者を選択する．
- 最良のピアの調査: ピアはリソース提供者に識別子の確認のための *Challenge* メッセージを送信し，正しい *Response* メッセージを受信した場合に次のフェイズに移行する．そうでない場合，一つ前のフェイズに戻る．
- リソースのダウンロード: リソースをダウンロードし，リソースを提供したピアの評価を更新する．

7.4.2 XRep

P2PRep の後継である *XRep* は，ピアの評判に加えてリソースの評判を保持する．まず，リソース選択と投票フェイズにおいて，ピアのリストの評判を尋ねる代わりに，リソースのリストとそのリソースを提供するピアの評判を尋ねる．次に，票の評価において，ピアの評判だけでなくリソースの評判を考慮し，要求元の優先順位に応じてダウンロードするピアとリソースを選択する．最後に，リソースをダウンロードした後，その質に応じてリソースの評価も更新する．

7.4.3 NICE における協力的なピアグループ

NICE 上に構築される**協力的なピアグループ** (*Cooperative Peer Groups*) では，優良なピアを識別し，グループを形成することで悪意あるピアを隔離する．各取引後にピアを評価し，スコアは**クッキー** (*cookie*) に保持される．

7.4.3.1 信頼評価

信頼グラフ (*trust graph*) において，有向エッジの始点ノードは信頼値を生成するノード，終点ノードはこれを受け取るノードであり，エッジの重みは始点ノードが終点ノードを信頼する度合いを表す．グラフ上で信頼を計算する方法は次の方法がある．

最も強いパス (*strongest path*)	最も強いパスは，二つのノードを繋いだパスのうち，パス上のエッジの最小値か，またはパス上のエッジの積が最大となるパスである．信頼値はそのパス上のエッジの最小重みとなる．
強い素なパスの加重合計 (*weighted sum of strongest disjoin path*)	信頼値は，全ての最も強い**素なパス** (*disjoint path*) の強さの加重合計となる．

7.4.3.2 信頼グラフ上の二つのノード間のパスの探索

ピアが他のピアの信頼性を評価するためには，そのピアまでの全てのパスを知っていればよい．ピアがパスを見つけるときは，クッキーから信頼できるノードを見つけ，全てのノードに探索リクエストを送信する．受信したノードが，もし探索するピアのクッキーを保持していない場合はリクエストを別の信頼ノードに転送し，保持している場合は要求元に返信する．

7.4.4 PeerTrust

PeerTrust では，ピアの評判を更に詳しく調べるために，ピアの評判の形成に次の 5 つの要因を用いる．

良好な取引の回数	他のピアとの良好な取引が多いほど，そのピアを信頼する．
取引の回数	悪意あるピアが，十分な良好な取引をした後に悪い取引をし続ける問題に対して，取引の総合回数を考慮する必要がある．
フィードバックの信頼性	もしピアが悪意がある場合，そのピアを信頼しないだけではなく，他のピアに対するそのピアのフィードバックも信頼しない．
取引内容	もし取引の中身を考慮しない場合，悪意のあるピアは小さい良好な取引を多く行い，大きい悪質な取引をいくらか行うことで，利益を稼げる．
コミュニティ内容	エージェントが取引のフィードバックを怠る場合に，フィードバックを行うピアに点数を与える例や，古かったり，よく知られていたり，先天的に信頼できるピアに対して信頼の計算で高い重みを加える例がある．

7.4.4.1 一般的な信頼の測定基準

信頼の基準は次の式で表される．

$$T(u) = \alpha \cdot \sum_{i=1}^{I(u)} S(u,i) \cdot Cr(p(u,i)) \cdot TF(u,i) + \beta \cdot CF(u)$$

ここで，$I(u)$ をピア u がシステム内の他のピアと行った取引の総数，i を取引の番号，$S(u,i)$ を i における u の相手の満足度合い，$Cr(p(u,i))$ を i における u の相手 p のフィードバックの評判，i における取引の要素を $TF(u,i)$，$CF(u)$ を u のコミュニティ内容の要因とし，α, β を正規化の係数とする．

7.5 個人評判と社会関係に基づく信頼システム

7.5.1 Regret

題材 s におけるピア i のピア j に対する**個人評判** (*individual reputation*) は，彼らの過去の取引に基づき，$R_{i\to j}(s)$ と表される．また，個人が社会に属すると，その振る舞いはその社会の他人の振る舞いに影響されるので，結果的に個人評判に加えて，**社会評判** (*social reputation*) を考慮する必要がある．ピア i, j が属する社会をそれぞれ I, J とすると，i から見た j の社会評判は次の式で表される．

$$SR_{i\to j}(s) = \xi_{ij} \cdot R_{i\to j}(s) + \xi_{iJ} \cdot R_{i\to J}(s) + \xi_{Ij} \cdot R_{I\to j}(s) + \xi_{IJ} \cdot R_{I\to J}(s)$$

ここで，$\xi_{ij}, \xi_{Ij}, \xi_{iJ}, \xi_{IJ}$ は $\xi_{ij} + \xi_{Ij} + \xi_{iJ} + \xi_{IJ} = 1$ を満たす係数である．さらに，様々な評判の側面を次の式でまとめることで，評判におけるオントロジーを形成する．

$$OR_{i\to j}(s) = \sum_{k \in \mathrm{children}(s)} w_{sk} \cdot OR_{i\to j}(k)$$

ここで，もし k が不可分な特徴の場合，$OR_{i\to j}(k) = SR_{i\to j}(k)$ となる．w_{sk} は総和が 1 になる重みである．

7.5.2 NodeRanking

7.5.2.1 ソーシャルネットワークの構築

ウェブのリンクや，メール，エージェントの取引などの情報から，有向グラフとしてソーシャルネットワークを構築すると，ノードからノードへの影響がノードからノードへの方向として反映される．エージェントの評判値は，他のエージェントが与える**参照** (*reference*) によって評価され，ソーシャルネットワークはグラフで描かれるため，エージェントの評判は単純に**入次数** (*incoming degree*) で測られる．ここで，ノード i の評判は i を参照するノードによって計算され，i が参照するノードは i の評判によって計算されるため，環状の参照がある場合，計算処理が終わらない可能性がある．これを解決するために，NodeRanking はランダムウォークの方法を採用する．

7.6 信頼管理

7.6.0.1 サーバベースの信頼管理

取引後に，各担当ピアが相手に関する見解をサーバに送信し，サーバはこれらのピアの信頼の管理と，これらのピアに関するクエリに返答する責任をもつ．

7.6.0.2 ゴシップベースの信頼管理

サーバを用いない信頼管理の方法は二つある．一つ目はシステム内のピア間で知識を交換するためにゴシップアルゴリズムを用いる方法で，二つ目は取引を行ったことがあるピアの局所的な評判のみを保持する方法である．知らないピアの評判を検索するときは，自身の隣接ノードからさらにその隣接ノードに対して次々と尋ね，この結果を局所的な知識と統合することでピアの信頼値を得る．

7.6.0.3 構造化 P2P ベースの信頼管理

ネットワーク上の各ピアに対し，その識別子はキーとされ，その評判はキーと共にネットワーク上にインデックス化される．ピアが他のピアの評判を検索するときは，探索キーとしてそのピアの識別子を用いたクエリを生成する．自身の評判の値を変更することが可能になるという問題に対して，ピアの評判をいくつかの場所に複製する方法がある．

7.6.1 XenoTrust

XenoTrust は，次の構成からなる *XenoServer Open Platform* 上で用いられる信頼管理システムである．

XenoServer	クライアントにホスティングサービスを提供する．サーバは *XenoCorp* に登録することでプラットフォームに参加でき，*XenoServer Information Service (XIS)* にそのサービスを広告する．クライアントがそのリソースを購入するときは，注文の検証と申し受けをサーバが XenoCorp に依頼する．
XenoCorp	サーバとクライアント両方における認証を請け負い，支払いの正確さを保証する．
XIS	XenoServer の状態とサービスのリストを保持する．これらの情報はクライアントと *Resource Discovery System* の両方から要求される．
Resource discovery system	クライアントが求める XenoServer を探すことを援助する．
クライアント	XenoServer からリソースを借りる．クライアントはシステムに参加するために自身を XenoCorp に登録し，Resource Discovery System か XIS を通して適合する XenoSever を探す．リソースを借りるために XenoCorp から依頼を購入し，最後にサーバでセッションを立ち上げ，タスクを配備する．

7.6.1.1 構造

XenoTrust は，XenoServer Open Platform 上の参加者の評判の検索と分散を行うイベントベースの信頼管理の構成部分である．クライアントは，各取引後に XenoCorp に情報を伝達することでサーバの評判に寄与し，また次の方法に沿ってサーバの評判を XenoCorp に問い合わせる．

広告の陳述 (*statement advertisement*)	ピアが他のピアの評判を報告するときに用いられる．陳述は，(**広告者** (*advertiser*)，**対象** (*subject*)，**トークン** (*token*)，値，タイムスタンプ）で形成され，広告者と対象は評判の評価者と受信者の識別子，トークンは評価の側面，値は評判のスコア，タイムスタンプは主張の有効期限を示す．
法則の配備 (*rule-set deployment*)	ピアの評判の問い合わせに用いられる．これは（**主役** (*principal*)，性質，広告者，関数，**トリガ** (*trigger*)）で形成され，主役は評判を調べるピアの識別子，性質は評判の側面，広告者は評価に考慮する広告者の集合，関数は評判を計算する方法，引き金はクライアントが通達を受信したときに値が変わる閾値の集合である．

7.6.2 EigenRep

ピア i から見たピア j の**局所評判** (*local reputation*) は，$s_{ij} = sat(i,j) - unsat(i,j)$ と表される．ここで，$sat(i,j)$ は i が j と行った良好な取引の数で，$unsat(i,j)$ は良好でない取引の数である．また，未知のピアを評価するために，**大域評判** (*global reputation*) を計算する必要がある．悪意あるピアが協力し，お互いの局所評判を高める問題を解決するため，局所評判をまとめあげる前に $c_{ij} = \frac{max(s_{i,j},0)}{\sum_j max(s_{ij},0)}$ で正規化をする．正規化された局所評判値は，**正規局所評判ベクトル** (*normalized local reputation vector*) $\vec{c_i} = (c_{i_1}, c_{i_2}, \ldots, c_{i_n})^{\mathrm{T}}$ に保存される．隣接ノードの局所評判も考慮すると，i から見た j の大域評判は，$t_{ij} = \sum_k c_{ik} \cdot c_{kj}$ で計算される．i の大域評判ベクトル $\vec{t_i}$ は，行列 $[c_{ij}]$ を C として，$\vec{t_i} = C^{\mathrm{T}} \cdot \vec{c_i}$ となる．ピアとピアの隣接ノードが未知のピアについての情報を持たない場合，ピアは隣接ノードの隣接ノードを考慮に入れる必要があり，上の式は $\vec{t_i}^{(2)} = (C^{\mathrm{T}})^2 \cdot \vec{c_i}$ と変形され，システム全体の知識は十分大きい n 回ステップ後に $\vec{t_i}^{(n)} = (C^{\mathrm{T}}) \cdot \vec{t_i}^{(n-1)} = (C^{\mathrm{T}})^n \cdot \vec{c_i}$ と計算される．

7.6.3 P-Grid による信頼管理

7.6.3.1 信頼測定

このシステムでは，他のピアで生成される**主張** (*claim*) の数と，他のピアについてのそのピアの主張の数に基づいてピアの評判が決まる．悪意のあるピアは高い主張の数を持つため，特定することが容易になる．P を全体のピアの集合，$c(p,q)$ を q についての p による評判とすると，ピアの評判は，$T(p) = \left|\{c(p,q)|q \in P\}\right| \times \left|\{c(q,p)|q \in P\}\right|$ で表される．

7.6.3.2 P-Grid ベースの信頼管理

ピアの**苦情** (*complaint*) は，キーとしてピアの識別子を用いることで保存することができる．この構造では，苦情は各ピアの保持するルーティングリンクに従って挿入，要求される．あるピアが他のピアと取引をし，その苦情を生成したいとすると，苦情のキーに従ってルーティングリンクを調べながらリクエストを転送する．