P2P ネットワークにおける評価管理のための EigenTrust アルゴリズム

1.導入
悪さをするピアがいるため、そいつらを排除したい。そのために、それぞれのピアに固有のグローバルな、全てのピアの経験を反映させた信頼指数を提案する。

2.設計思想

1.システムは中央権力ではなく、自己防衛する仕組みにしなければならない
2.システムは匿名でなければならない。
ピアの評価は、例えばピア自身の IP アドレスのような、外部的に関連付けられたものではなく、不透明な識別子に関連付けなければならない。
3.システムは、新しい参加者に評価を割り当ててはならない。つまり、評価はいつくかのトランザクションにおける継続したいい行いによって得られるべきであり、低い評価の悪意のあるピアが彼らの不透明な識別子を変え、新しい参加者の地位を得るのに有利であってはならない。
4.システムは計算や構造、貯蓄、メッセージの複雑性において、最小限のコストでなければならない。
5.システムはお互いに知っていて、システムを転覆させようとする集団に対して堅牢でなければならない。

3.評価システム

成功している評価システムの重要な例として、オンラインオークションシステムである、eBay が挙げられる。eBay では、買い手と売り手はそれぞれのトランザクションの後、お互いを評価できる。そして全ての参加者の評価は、六か月にわたるこれらの評価の合計である。これは、中央集権的システムに依存している。

分散型システムにおいても、ピアは eBay と同じようにトランザクションごとにお互いに評価できる。
例えば、おのおのの時刻において、ピア i が j からファイルをダウンロードするとき、正$(tr(i,j)=1)$または負$(tr(i,j=-1))$としてトランザクションを評価できる。
eBay と同じように、s_{ij}を i が j からダウンロードしたそれぞれのトランザクションの評価の合計として定義する。

$$s_{ij} = \sum tr_{ij}$$

以前のシステムは、少ないピアの評価しか集計しない、あるいはすべてのピアの評価を集計し、メッセージ数が多くなりすぎると言う欠点があった。

私たちのアプローチは、"transitive trust"の考え方に基づいている。ピアｉは、自身に信頼できるファイルを提供したピアに対し、高い評価を持つ。さらに、ピアが提供するファイルに正直であるならば、ローカルな信頼度の報告にも正直であるとみなされる。
メッセージの複雑性が有界であることを証明し、かつ経験的に小さく、少ないコードでいかにこれを行うかを示す。

4.固有信頼度(EIGENTRUST)
EIGENTRUST アルゴリズムを解説する。それぞれのピアｉのグローバルな評価は、他のピアによりピアｉに割り当てられたローカル信頼値により与えられる。それらの他のピアは、わりあてられたピアによるグローバルな評価により重みづけられる。
4.1 では、エレガントな確率解釈へと導く、ローカル信頼度を正規化する方法と、これらの値を集計する効率の良いアルゴリズムを示す。
4.2 では、正規化された信頼度を、賢いやり方で集計する方法を議論する。
4.3 では、ローカルあるいはグローバル信頼度の確率解釈を議論する。
4.4～4.6 では、グローバル信頼度を計算するアルゴリズムを示す。

4.1 ローカル信頼度の正規化

ローカル信頼度を集計するためには、それらを同じ方法で正規化する必要がある。
Normalized local trust value （正規化されたローカル信頼度）,c_{ij}を以下の様に定義する。

$$c_{ij} = \frac{max(s_{ij}, 0)}{\sum_j max(s_{ij}, 0)}$$

$0 \leq c_{ij} \leq 1$が保証される。（分母が 0 になるときは 4.4 で述べる。）
この方法にはいつくかの欠点がある。
まず、ピアｉと交流のある、あるいは過去に低い評価をしたピアを見分けることができない。
次に、これらのc_{ij}は関連づいていて、絶対的な解釈がない。もし$c_{ij} = c_{ik}$ならば、ピアｉからのそれぞれの評価が同じであることしかわからず、両方非常に信頼できるのか。それとも平凡なのかわかならない。
しかしこれらの欠点にも関わらず、実質的にいい結果を達成できる。

この方法により、それぞれの反復におけるグローバル信頼度の再正規化を行うこと（これは大きな分散化された環境においてとてつもなくコストが重い）なしに以下に述べる計算を実行できる。

4.2 ローカル信頼度の集計

ローカル信頼度の正規化を集計したい。自然な方法としては、ピア i のために、交流のある他のピアに、他のピアに関する評価を伺うことだ。そして彼らの評価を、ピア i を信頼しその評価で重みづけることは道理にかなう。

$$t_{ik} = \sum_j c_{ij} c_{jk}$$

t_{ik}は、ピア i が友人たちに訊いたことに基づいた、ピア k の信頼度を意味している。
これは行列によっても表記できる。c_{ij}を行列Cによって表し、t_iをt_{ik}の値を含むベクトル（論文の説明が分かりにくいので書き足すと、t_{ik}の i 行目を抜き出し、縦ベクトルにしたもの）とすると、$t_i = C^T c_i$となる。（$\sum_j t_{ij} = 1$が満たされていることに注意。）
これはそれぞれのピアがそれ自身の経験値より広いネットワークの知見を得るために有効である。しかし、ピア i に保存された信頼値はピア i とその友人たちの経験値しか反映していない。さらに広い知見を得るため、ピア i は友人の友人に訊くだろう$(t = (C^T)^2 c_i)$。同様に続ければ$(t = (C^T)^n c_i)$となり、n が十分大きい反復により、ネットワークの完全な知見を得る。（Cが既約かつ非振動の条件のもと、これは我々が 4.5 で説明する。）
幸運なことに、n が大きい場合、信頼度ベクトル t は全てのピア i に対して同じベクトルに収束する。つまり、それはCの左固有ベクトル（C^Tの固有ベクトル）に収束する。言い換えれば、t はこのモデルにおけるグローバル信頼度である。この成分t_jは、システムがどれぐらいピア j を信頼するかを定量化する。

4.3 確率解釈

本方法における、ストレートな確率解釈が存在することを示すことは有効である。もし友人たちが信頼できるピアを探しているならば、以下に述べる規約を使ってネットワークを探索できる。それぞれのピア i がピア j に確率c_{ij}の確率で探索する。しばらく後、友人たちは信頼できないピアよりは、信頼できるピアを探索していると推測される。Cにより定義されるマルコフ連鎖の定常的な収束地点は、グローバル信頼度ベクトルtである。

アルゴリズム 1

$t^0 = e$

$repeat \quad \mathrm{t}^{\mathrm{k}+1} = C^T t^k$

$until \left|\left|\mathrm{t}^{k+1} - \mathrm{t}^{\mathrm{k}}\right|\right| < \exists\epsilon$

4.4 基本的な EigenTrust

ここでは、今のところは P2P ネットワークの分散化された特徴を無視し、基本的な EigenTrust アルゴリズムを説明する。いくつかの中央サーバーが全てのc_{ij}の値を知っていて、計算を実行すると仮定する。4.6 では、分散環境における計算方法を説明する。

eを m 個のピアたちにおける一様確率分布を表す m 次元ベクトルと定義したうえで、我々は単純に、n が大きい場合に$t = (C^t)^n e$を計算したい。（4.2 では$t = (C^t)^n c_i$を計算したいと言ったが、いずれにせよ左固有ベクトルに収束するので e を代わりに使う。）

最も基本的な水準においては、アルゴリズム 1 を採用する。

4.5 実用上の話題

このアルゴリズムにおいては言及されていない、3 つの実用上の話題がある。信頼度の先見的評価、非活動的なピア、悪意のある集団である。

信頼度の先見的評価

ネットワークにいくつかの、信頼できるピアがいる場合がある。例えば、信頼に値する最初にネットワークに参加したごく少数のピア、設計者や P2P ネットワークの初期からの利用者は、創ったネットワークを破壊しようとする気持ちは少ないだろう。それらの評価を自然で調和した方法により合体させるのは有効だろう。

これを、あらかじめ信頼されたピアに基づくいくつかの分布pを定義することにより行う。

例えば、いくつかのピアの集合 Pが信頼できるとわかった場合、$\left(p_i = \frac{1}{|P|}, if\ i \in P, p_i = 0\ otherwise\right)$により定義できる。この$p$を 3 つの方法で用いる。まず、悪意のあるピアの存在するとき一般に、$t = (C^T)^n p$は$t = (C^T)^n e$よりも早く収束するから、pを初期ベクトルとして使う。

非活動的なピア

もしピア i が他の誰からもダウンロードしない、もしくはほかの全てのピアに 0 を割り当てた場合、c_{ij}は定義されない。この場合、c_{ij}はp_jで設定する。つまり、ピア i が他の誰も知らないあるいは信頼しない場合には、あらかじめ信頼されたピアを採用すればよい。

悪意のある集団

悪意のある集団とは、お互いに知っていて、ローカルに高い信頼度をお互いに与え、システムを脅かすために他のピアには低いローカル信頼度しか与えない、高いグローバル信頼度を持つ悪意のあるピアの集団である。
この問題について、この式を議論することで言及する。

$$t^{k+1} = (1-a)C^T t^k + ap$$

aは 1 より小さい定数である。これは、それぞれのピアにピア P の信頼度を持たせることで、集団を破棄するため、全てのピアの評価ベクトルを$c_i = (1-a)c_i + ap_i$
とすることと等価である。確率論的に、これは 4 章で与えられた確率論的モデルによりネットワークを探索する代理者が、悪意のあるピアの探索に失敗しにくいという主張と等価である。なぜならばそれぞれのステップで、代理者があらかじめ信頼されたピアを探索する一定以上の確率があるからである。これは行列 C が既約かつ非振動的であり、計算が収束へ向かうことを保証していることに気付いてほしい。
定式化されたアルゴリズム 2 を示す。

アルゴリズム 2
repeat
$t^{k+1} = C^T t^k$
$t^{k+1} = (1-a)t^{k+1} + ap$
until $\left|\left|t^{k+1} - t^k\right|\right| < \epsilon$

あらかじめ信頼されたピアは収束を保証し、悪意のある集団を破壊するゆえ、このアルゴリズムにおいて不可欠であることを強調しなければならない、
それゆえ、あらかじめ信頼されたピアの選択、とりわけあらかじめ信頼されたピアが悪意のある集団のメンバーでないことは重要である。システムはごく少数のあらかじめ信頼されたピアを選ぶこともできる（例えばネットワークの設計者)。あらかじめ信頼されたピアの様々な方法の完全な調査も、面白い研究領域であるが、本論文では言及しない。

4.6 分散 EigenTrust
ここで、我々は全てのピアがネットワークで協力し、グローバル信頼度の計算や記憶を行う、メッセージのコストが最小限になるアルゴリズムを示す。
分散環境で最初に挙がる挑戦的課題は、Cとtをいかに記憶しておくかである。前章では、それぞれのピアはそれぞれのローカル信頼ベクトルc_iを保存できると提案した。ここでは、それぞれのピアが自分自身のグローバル信頼度ベクトルt_iも保存することも提案する。(安全性の問題は、5 章で述べる。)
実際、それぞれのピアは自己のグローバル信頼度を計算できる。

$$t_i^{k+1} = (1-a)\left(c_{1\mathrm{i}} t_1^k + \cdots + c_{n\mathrm{i}} t_n^k\right) + ap_i$$

これは、$t^{k+1} = (1-a)C^T t^k + ap$を成分ついて展開した表式である。ピア i は他のピアと限られた交流しかないため、表式の成分の多くは 0 になることに気付いてほしい。これはアルゴリズム 3 で示される単純な分散アルゴリズムおいて役立つ。ここで、２つの興味深いことがある。
まず、あらかじめ信頼されたピアしかそれらのpを知る必要がない。これはあらかじめ信頼されたピアは匿名にできる。つまり他の誰も、それらがあらかじめ信頼されたものであると知る必要がない。（あらかじめ信頼されたピアは高い信頼度を持つため、特定しうると思うかも知れない。しかし、シュミレーションによれば、あらかじめ信頼されたピアはtの平均を上回るが、t_iの最大値をとることはあまりない。）
次に、多くの P2P ネットワークでは、それぞれのピアは他のピアと限定された交流しか持たない。これには利点が２つある。まず、ほとんどのc_{ij}は 0 であるから、$t_i^{k+1} = (1-a)\left(c_{1\mathrm{i}} t_1^k + \cdots + c_{ni} t_n^k\right) + ap_i$の計算が重くならない。次に、$A_i$と$B_i$が小さいから、送受信されるメッセージ数が少ない。ネットワークにとても活動的なピアが多くなった場合は、それぞれのピアが報告できるローカル信頼度c_{ij}を制限すればよい。

アルゴリズム３：分散 EigenTrust アルゴリズム
定義
A_i:ピア i からファイルをダウンロードされたピアの集合
B_i:ピア i がファイルをダウンロードしたピアの集合

アルゴリズム
それぞれのピアがこれを行う｛
$j \in A_i$なる全てのピアに、$t_j^0 = p_j$に設定したか訊く
repeat
$t_i^{k+1} = (1-a)\left(c_{1\mathrm{i}} t_1^k + \cdots + c_{ni} t_n^k\right) + ap_i$を計算する
$c_{ij} t_i^{k+1}$を$j \in B_i$なるすべてのピアに送る。
$\delta = \left|\left|t^{k+1} - t^k\right|\right|$を計算する
$j \in A_i$なる全てのピアから$c_{ij} t_i^{k+1}$が返るのを待つ
until $\delta < \exists\epsilon$

4.7 アルゴリズムの複雑性

アルゴリズムの複雑性について２つの側面から見る。
まず、アルゴリズムの収束が速い。1000 ピアの 100 サイクルの場合（いかにシステムを

シュミレートしたかは 7.1 で述べる。)、図 1 で、残渣$\left\|t^{k+1}-t^{k}\right\|_{1}$(一乗ノルム)を示す。

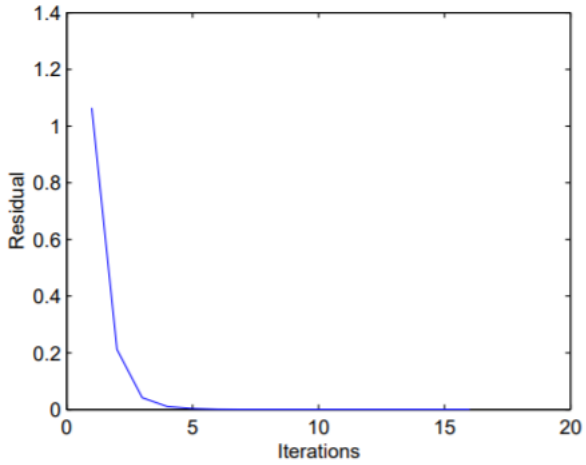

Figure 1: EigenTrust convergence

明らかに、アルゴリズムは 10 回の反復以下で収束している。分散アルゴリズムでは、これはピア間の 10 回未満の信頼度指数の交換に対応する。この理由は論文 10 に示す。
次に、ピアのローカル信頼度の報告回数は厳密に制限できる。EigenTrust の定式化では、それぞれのピアはローカル信頼度の全ての集合の部分集合を報告する。予備シュミレーションでは、この方法が、ここで示した、全てのローカル信頼度を報告するアルゴリズムと同程度であることが示された。

5.安全な EIGENTRUST

前章で示したアルゴリズムでは、それぞれのピア i は自己の信頼度t_iを計算、報告した。悪意のあるピアは簡単に嘘の信頼度を報告でき、システムを転覆できうる。
これに対し、２つのアイデアを実装することで対処する。
まず、現在の信頼度はピアそれ自身に計算あるいは保存させてはならない。ネットワークには信頼度を計算するための異なるピアを置く。
次に、悪意のあるピアが他のピアの信頼度を計算できるとき、間違った結果を返しかねないから、ネットワークにおけるピアの信頼度は他の少なくとも一つ以上のピアに計算してもらう。
分散アルゴリズムの安全版では、M 個のピアがピア i の代わりに値を計算する。ピアがピア i の信頼度を知りたいとき、M 個の管理者に訊くことで、多数派は悪意のあるピアの不完全性を示すことができる。
管理者に割り当てるには、CAN や Chord などの分散型ハッシュ表（distributed hash table DHT）を用いる。DHT は、例えばファイル名からベクトル空間の座標への関数のよう

な、決定的にキーをマップするハッシュ関数を用いる。すべてのピアでベクトル空間の領域を覆うために、ベクトル空間がダイナミックに分割されることもある。ピアは(キー、値)の組み合わせを記憶しておく責任がある。
我々のアプローチにおいて、ピアのスコア管理者は IP アドレスと TCP ポートから DHT ハッシュ空間の座標へのような、ピア特有の ID をハッシュすることで位置が分かる。現在この座標をカバーしている DHT の一部分が、スコア管理者として指定される。すべてのピアはこのようにしてスコア管理者の位置がわかる。スコア管理者により実行できるように、最初のアルゴリズムを修正する。

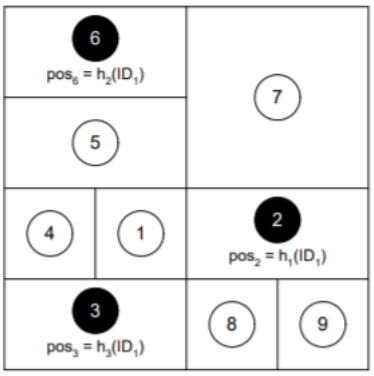

Figure 2: Two-dimensional CAN hash space

例として、図 2 の CAN を考える。ピア 1 の固有 ID,ID_1はピア 2.3,6 にカバーされた座標に、それぞれハッシュ関数$h_1.h_2,h_3$によりマップされている。それゆえにこれらのピアはピア 1 のスコア管理者である。
P2P システムに固有のダイナミクスに対処するため、うまく設計された DHT に頼ることになる。例えば、スコア管理者がシステムから離脱するとき、この部分を抜かして DHT ベクトル空間の隣のピアへと通り過ぎることになる。DHT はデータ（この場合は信頼度）の損失を防ぐため、データの複製を導入する。

5.1 アルゴリズムの説明

ここで、グローバル信頼度ベクトルを計算するための安全なアルゴリズムを説明する。
以下の定義を用いる。:
各々のピアには M 個のスコア管理者があり、その DHT の座標は、一方的で安全なハッシュ関数$h_0,h_1,\dots,h_{M-1}$をピア固有の識別子として適用する。
pos は、ハッシュ空間におけるピア i の座標である。それぞれのピアはまたスコア管理者でもあるので、子の集合D（信頼度の計算がこのピア i により行われるピアも含まれる）も割り当てられる。スコア管理者として、ピア i はその子ピアd$(d \in D_i)$の評価ベクトルc_d^iを持つ。

さらに、その子ピア d からファイルをダウンロードしたピアの集合である、A_d^iから信頼度の情報をピア i は得る。最後に、子ピア d がファイルをダウンロードした他のピアの集合である、B_d^iからピア i は情報を得る。グローバル信頼度の計算に基づき、その子ピア d は他のピアの信頼度の評価をスコア管理者に報告する。

アルゴリズム 4 ：安全な EigenTrust アルゴリズム

それぞれのピア i は、以下を行う

ローカル信頼度を$h_m(pos_i), m = 1, \dots M-1$のスコア管理者に報告
$d \in D_i$なる子ピアdと集合B_d^iのローカル信頼度を集める
子ピアdのローカル信頼度c_{dj}をスコア管理者$h_m(pos_i), m = 1, \dots M-1$
に報告する
子ピアのA_d^iのローカル信頼度を集める。
それぞれの$d \in D_i$なる子ピア d は以下を行う

$j \in A_d^i$なるすべてのピアに$c_{jd}p_j$を訊く
repeat
$t_i^{k+1} = (1-a)\left(c_{1i}t_1^k + \cdots + c_{ni}t_n^k\right) + ap_i$を計算する
$c_{ij}t_i^{k+1}$を$j \in B_i$なるすべてのピアに送る
$\delta = \left|\left|t^{k+1} - t^k\right|\right|$を計算する
$j \in A_i$なる全てのピアから$c_{ij}t_i^{k+1}$が返るのを待つ
until $\delta < \exists\epsilon$

匿名性
あるピアが、特定の座標にあるピア信頼度を計算するピア ID を知るのは不可能であるゆえ、悪意のあるピアは他の悪意のあるピアの評価を減らせない。

無作為性
システムに参加するピアは、ハッシュ空間のどの位置にいたいかを選ぶことはできない（これがうまく設計された DHT の利点である）ゆえ、ピアが、例えばそれ自身の ID を計算したり、そのために正確にハッシュ空間の位置を指定することは不可能である。

冗長性
複数のスコア管理者が一つのピアの信頼度を計算する。一つのピアに複数の管理者を割り当てるため、いくつかの多次元ハッシュ関数を用いる。システムにおいてピアは座標空間の特定の領域を占有する。ピアの固有 ID は、これゆえにおのおのの多次元ハッシュ空間

の異なる点へマップされる。

5.2 議論

ここで、２つ重要な指摘すべき点がある。

まず、P2P ネットワークにおける安全なスコア管理の問題は、潜在的に評判管理や報酬制度、P2P マイクロペイメント、その他諸々にとって重要である。この仕事は主には、コアな EigenTrust アルゴℜズムに貢献する。いくつかの安全なスコア管理は EigenTrust アルゴリズムに必須であるから議論するが、コアな EigenTrust アルゴリズムはさらに多くの異なる安全なスコア管理の体系を用いていることを忘れてはならない。

次に、ここで提案された安全プロトコルとは、いかに大きな集合を小さくする、あるいはプロトコルの小分けにして取り扱うかである。これらのプロトコルは、伝統的なやり方では安全にならない。それに代わり、ピアが間違ってスコアを報告する確率を小さくすることを我々は示せる。このことについては、[20]がより詳しい。

6.グローバル信頼度の利用

P2P システムにおいてグローバル信頼度を使う明確な理由が２つある。

１番目に、利用者を信頼できるピアからダウンロードするように仕向けることで、悪意のあるピアを孤立させることだ。2 番目は、信頼できるピアに報酬を与えることで、ピアが積極的にファイルを共有できることだ。

悪意のあるピアの隔離

システムは、信頼度t_jを用いて利用者により信頼できるピアからダウンロードさせることができる。これを行う一つの方法は、それぞれのピアに最も信頼できるピアのみからダウンロードさせることだろう。しかしそうすれば 7 章で示すように、そのピアはダウンロードの負荷がかかりすぎるし、さらには、評価はファイルの共有の上に成り立っているのに、新しい参加者はシステム上で評価を受けることができない。

もう一つの方法は、信頼度に基づき確率的にダウンロード先を選ばせることだ。特に、ピア j の信頼度t_jに比例する確率でダウンロードさせることができる。

そのような方針により、ネットワークにおけるダウンロードをバランスさせ、新しい参加者も評価をうけることができるようにしながら、ダウンロード数が停滞しないようにできる。これを、7 章の実験によって検証する。

ピアが、グローバル信頼度と自身のもつ他のピアに関するローカル信頼度の組み合わせによって、（さらに$t_{personal} = dt + (1-d)c$,$d$は 0 から 1 の定数、なる信頼度ベクトルを用いて）簡単にダウンロード先の選択を選べることは重要である。この方法においては、１つ

のピアだけではなく、さらにはネットワークの他のピアさえも、悪いサービスをしたピアからのダウンロードを避けることができる。

ただ乗りに共有させること
次に、システムは高い信頼度を持つピアに報酬を与えることができる。例えば、他の信頼できるピアとたくさんのつながりを持つ信頼できるピアに報酬を与えることができる。強く信頼されているピアに報酬を与えることは一石二鳥である。
まず、信頼できるファイルを共有することでしかグローバル信頼度を上げることはできないから、利用者にファイルの共有を促すことになる。最近の gnutella ネットワークでは、7%のピアが 50%以上のファイルに関する責任を負っていて、25%のネットワーク上のピアは全くファイルを共有していない。信頼度に基づいた促進により、P2P ネットワークのただ乗りを防ぎうる。
次に、強く信頼されているピアに報酬を与えることにより、悪意のないピアが、悪意のあるピアから偶然にダウンロードされた信頼性のないファイルを削除することを促し、ネットワークを整頓させる。

7.実験

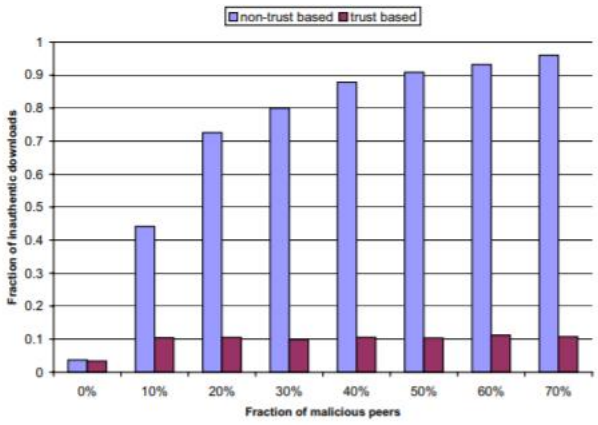

Figure 5: Reduction of inauthentic downloads by basing download source selection on global trust values in a network where independent malicious peers are present. Upon activation of our reputation scheme, the number of inauthentic downloads in the network is significantly decreased to around 10% of all downloads in the system, malicious peers in the network are virtually banned from uploading inauthentic files.

図 5 ：
縦軸：不正なダウンロードの割合
横軸：悪意のあるピアの割合
青：信頼度に基づいていない

紫：信頼度に基づいている
独立した悪意のあるピアのいるネットワークにおいて、ダウンロード資源の選択をグローバル信頼度に基づいて行うことにより、信頼できないダウンロードを削減している。我々の評価法の下では、ネットワークにおける信頼できないダウンロードは全てのダウンロード数の 10%前後と顕著に減少していて、ネットワーク上の悪意のあるピアは信頼できないファイルをアップロードすることを事実上禁止されている。

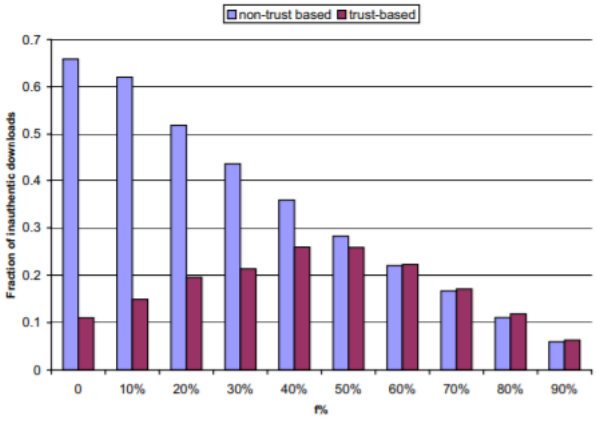

Figure 7: Trust-based reduction of inauthentic downloads in a network where a fraction of peers forms a malicious collective and returns authentic files with certain probabilities. When malicious peers partly provide authentic uploads, they receive more positive local trust values and will be selected as download sources more often, also increasing their chances to upload inauthentic files. Yet, uploading authentic files may be associated with a cost for malicious peers.

図 7 ：
縦軸：不正なダウンロードの割合
横軸：f%（論文中では、悪意のあるファイルをアップロードする確率）
青：信頼度に基づいていない
紫：信頼度に基づいている
信頼度に基づくことによる、一定の確率で信頼できないファイルを返す、悪意のあるピアがいるネットワークにおける悪意のあるファイルのダウンロードの削減。悪意のあるピアが部分的に信頼できないアップロードを行うとき、正のローカル信頼度を得るため、よりダウンロード源として選ばれ、彼らの悪意のあるファイルをアップロードさせる機会を増やしてしまう。しかし、悪意のあるファイルをアップロードすることは悪意のあるピアにとってのコストとなる。
（以下省略、シュミレーションの方法や結果など）