P2P システムにおけるアクセス履歴を用いたピア推薦に関する研究

糸川　剛[†1]

大規模情報処理システムのバックボーンとして Peer-to-Peer(P2P) システムが着目されている．P2P システムの一例である分散ハッシュテーブルを用いたアプリケーションシステムを考えるとき，格納されるオブジェクトの特徴に応じて特定グループのピアによる検索が多くなる場合が考えられる．この場合，システム全体を検索する前に，特定グループのピア同士が互いのキャッシュを参照することで，より高速なオブジェクトの取得が可能になると考えられる．本稿では，分散ハッシュテーブルの例として Chord を取り上げ，各ピアが検索処理に関わったときに，検索対象となったオブジェクトと検索を行ったピアを履歴として記録し，その履歴をもとにピアに対して別のピアを推薦する手法の検討を行う．

A Consideration of Peer Recommendation using Local Access Log on P2P Systems

Tsuyoshi Itokawa[†1]

In this paper, we consider peer recommendation method for making peer groups on application systems based on P2P distributed hash tables. Our method simply adds an access log data structure on each peer, and each peer sends peer recommendation messages to peers referring the data structure. Under the case that some objects tend to be accessed frequently by peers with common interest, we think that forming groups of such peers using peer recommendation messages makes the systems efficient. In our experiments we evaluate the accuracy of our peer recommendation method varying some parameters.

†1 熊本県立技術短期大学校
Kumamoto Prefectural College of Technology

1. はじめに

コンピュータの低価格化やインターネットへの高速接続環境の普及にともないインターネットに接続されるコンピュータ数は近年爆発的に増加した．ネットワーク接続された所有者の異なる多数のコンピュータの資源，特に記憶領域を統合して有効活用する手段として P2P(Peer-to-Peer) システムによる分散ファイルシステムの研究が活発に行われている[1)]．P2P システムによる分散ファイルシステムでは，どのようなオブジェクトを管理するかにより様々なサービスが考えられるが，ウェブアクセスの高速化のためのキャッシングシステムであるウェブキャッシングシステム，動画配信サービスのバックエンドシステムなど，オブジェクトにアクセスする人の持つ嗜好や地域などによりオブジェクトへのアクセス傾向に偏りがある場合がある．このような場合，例えば共通の地域や嗜好ごとのグループを構成し，システム全体を検索し世界中に分散配置されたオブジェクトにアクセスする前に，グループ内に求めるオブジェクトが存在するかどうかを確認することで，目的オブジェクトの取得にかかる時間を短縮することができると考えられる．

ところが，P2P システムの全体を管理するノードが存在しないピュア P2P システムにおいては，システムを構成するピア全体についての情報も一元管理されていない．そのため，ピアが存在する地域，ピアの持つ嗜好といったような情報を入手することが困難であるので，同じ嗜好を持ったピアでグループを形成することは容易ではない．本研究では，P2P システムによる分散ファイルシステムの例として分散ハッシュテーブルを取り上げ，分散ハッシュテーブルにより管理されるオブジェクトにはビットベクトルで表現される特徴値が定まっていると仮定する．この仮定のもと，分散ハッシュテーブルを構成する各ピアが，オブジェクト検索過程で携わった処理についてアクセス履歴を記録し，そのアクセス記録を用いてあるピアに対して別のピアとグループを組むことを推薦する手法についての検討を行う．

以下，2 章では本研究の位置づけについて述べ，3 章で分散ハッシュテーブルの具体例として Chord を用いて提案手法についての説明を行う．4 章において，提案手法によりどの程度の正確さで正しくピアの推薦が行われるかをシミュレーションにより確認し，5 章においてまとめと今後の課題について述べる．

2. 関連研究

ネットワーク接続されたコンピュータが各々自律的に振る舞い，全体としてシステムを成す P2P システムがある．P2P システムの中でも全体を管理するための情報を保持する特

別なサーバが存在せず，完全な分散構成であるピュア P2P システムは，各ピアの相互接続構成のなりたちにより構造 (structured) 型ネットワークと非構造 (unstructured) 型ネットワークに分類される．構造型ネットワークでは各ピアがある数学的な規則より実際のネットワーク接続のトポロジの上位層において相互接続されるオーバーレイネットワークを構成する．構造型オーバーレイの例としてハッシュ法のハッシュテーブルを各ピアに分散配置する分散ハッシュテーブル (DHT, Distributed Hash Table) があり，その実現法として Chord[5],CAN[4] などの手法が提案されている．

分散ハッシュテーブルにおいてどのようなオブジェクトを管理するかにより様々なアプリケーションサービスが考えられるが，その一例として Web アクセスのキャッシュを保持する Web キャッシングシステムがある．Iwamaru ら[3] により，P2P を用いた Web キャッシングシステムにピアのグループを組んで参加した場合，検索性能が向上することが示された．この研究においては，ユーザが複数台のコンピュータ資源を持つ場合，それらをピアのグループとして Web キャッシングシステムに参加することを想定しており，グループは事前に確定しているものとしている．本研究で検討を行うピア推薦手法で，1 台で参加したピアもグループを構成することができるならば，この Web キャッシングシステムの性能向上が見込まれる．

本研究は，分散ハッシュテーブル上にオブジェクトが何らかの数学的規則にしたがって配置されるという前提の下，システム全体を総括する管理情報ではなく，各ピアが通常のオブジェクト検索のための lookup 処理を行う過程において記録する情報をもとに，別のピアに対して同じ嗜好を持つであろうピアを推薦することを目的とする．特定の分散ハッシュテーブルによらないピアの推薦を志向しているが，本稿では例として Chord を用いて実現した分散ハッシュテーブルのもとで推薦手法の詳細を記述し、性能評価のシミュレーションを行うので，以下，Chord に関する基本的なデータ構造とアルゴリズムの解説を行う．

Chord において分散ハッシュテーブルを構成する各ピアは SHA-1 などのハッシュ関数にしたがって m ビット長の識別子が与えられ，2^m を法とするリング状の識別子空間にマッピングされる．各ピアは自身の次に大きな識別子を持つピアへのポインタ successor，および，ピアの参加／離脱に備えるため，自身の次に小さい識別子を持つピアへのポインタ predecessor を保持する．すなわち，分散ハッシュテーブルを構成する各ピアは successor, predecessor により識別子の順で連結された双方向環状リストを構成する．また，管理対象となる各オブジェクトは自身のキー値を元に，ピアの識別子を決定したのと同様の手法で識別子が与えられ，その識別子以上の値を持つ最初のピアの管理下に置かれる．

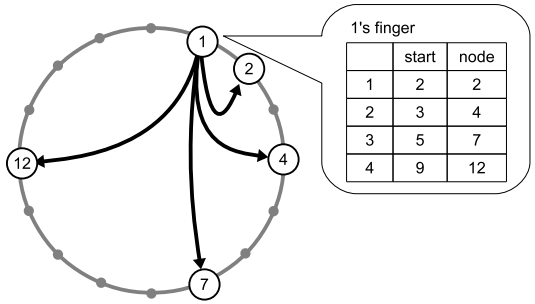


図 1 ノード 1 の持つフィンガーテーブル

リング状の識別子空間のことを以後 Chord リングと呼び，Chord リング上で目的の識別子の管理を行うピアへ効率よく到着するため，各ピアはフィンガーテーブルと呼ばれるサイズ m のテーブルを持つ．フィンガーテーブルの i 番目のエントリ finger[i](ただし，$1 \leq i \leq m$) は，finger[i].start と finger[i].node の二つのフィールドから構成され，識別子 n を持つピアにおいては，finger[i].start は $n + 2^{i-1} (\mathrm{mod} 2^m)$ の値，finger[i].node は Chord リング上で finger[i].start 以上の識別子を持つ最初のピアへのポインタを格納する．ここで，ピアへのポインタと識別子は本来別の値（たとえばポインタは IP アドレス，識別子は IP アドレスなどを元に決められたハッシュ値）であるが，簡単のため以後の説明においては両者を同一視する．

以上のデータ構造の下，ピア n に識別子 k の問い合わせである lookup(k) が到着した場合，k が n 自身の責任範囲 (n.predecessor, n] 内にあれば問い合わせに n 自身が回答し，そうでない場合，自身のフィンガーテーブルを参照し，k が [finger[i].start, finger[$i+1$].start) の範囲内にあれば，finger[i].node へ lookup(k) のルーティングを行う．ただし finger[$m+1$].start には仮想的に n が入っているものとする．

図 1 に $m = 4$ の Chord リング上に識別子が 1,2,4,7,12 であるようなピアが配置された場合の識別子 1 のピアが持つフィンガーテーブルを示す．識別子 1 のピアが識別子 6 のオブジェクトの lookup 処理を行う場合，$6 \in [5, 9)$ であるので，フィンガーテーブルの finger[3].node を参照し識別子 7 を持つピアにこの lookup をルーティングする．識別子 7 を持つピアでは，predecessor が 4 であり，$6 \in (4, 7]$ であるので，自身の責任範囲内にあるので，問い合わせに回答するための処理を行うこととなる．

3. 提案手法

Chord により実現する分散ハッシュテーブルをもとに，本稿で提案するピア推薦の手法について説明を行う．

基本的なデータ構造については Chord を継承するが，各ピアが過去に携わった lookup の中継履歴を保持するためのデータ構造を追加する必要がある．このためのデータ構造 accesslog は，オブジェクトの検索を行ったピア，すなわち lookup の起点となったピアの識別子とキーとし，そのピアがそれまで検索を行ったオブジェクトの識別子の集合を得る写像とする．例えば，あるピアが,「識別子が p_1 のピアが識別子が k_1, k_2, k_3 である三つのオブジェクトをこれまでに検索した」ことを lookup 処理に関わったために記録している場合には，そのピアの accesslog には $(p_1, \{k_1, k_2, k_3\})$ のエントリが格納されている．以後，accesslog に対してピアの識別子をキーとしオブジェクトの識別子集合を得る手続きを accesslog.get と表記する．例えば，accesslog.get(p_1) の実行結果として集合 $\{k_1, k_2, k_3\}$ を得る．また，accesslog に記録されているピアの識別子の集合を得る操作を accesslog.keyset() とする．

さらに，オブジェクトの識別子をキーとしオブジェクトの特徴ビットベクトルを記録するデータ構造 features を追加する．「識別子が k_1 のオブジェクトは特徴ビットベクトル $010\cdots011$ を持つ」といった情報を記録するために features に $(k_1, 010\cdots011)$ というエントリを格納する．以後，features に記録されている特徴ビットベクトルを得る手続きを features.get と表記する．例えば，features.get(k_1) の実行結果としては特徴ビットベクトル $010\cdots011$ が得られる．

これらの追加のデータ構造を用い，ピア $srcPeer$ が識別子 $targetObject$ のオブジェクトを検索するための処理 lookup($targetObject$,$srcPeer$) がピア n に到着した際の lookup の手順は次の通りである．

これまで記録した accesslog に記録されている各ピア p に関し，p が検索したすべてのオブジェクトの特徴ビットベクトルと $targetObject$ の特徴ビットベクトルとの類似度を計算する．二つのオブジェクトの特徴ビットベクトル b_1, b_2 の類似度はハミング距離をもととした，次の式 (1) により計算するものとする．

$$\text{similarity}(b_1, b_2) = (b_1.\text{size} - \text{card}(b1 \oplus b2))/b_1.\text{size} \quad (1)$$

ただし，card はビットベクトルの 1 の数を数え上げる手続きであり，b_1, b_2 は長さが等しく，b_1.size($= b_2$.size) はその長さを表すものとする．類似度は二つのビットベクトルが同じ場合 1，すべての要素が異なる場合に 0 となる．ピア p が検索したすべてのオブジェクトについての平均類似度を計算し，記録する．

accesslog に記録されている各ピア p について求めたピア p が検索したオブジェクトの平均類似度をもとにして推薦を行う．現時点では単純にこの平均類似度がしきい値 β を超えた場合に推薦を行うものとする．N ピアからなる Chord システム上で 1 回の検索の lookup 処理に関わるピアの数は最大 $\log N$ であり，システム全体で総数 L 回の検索が行われた場合には，lookup 処理に関わるピアの総数は最大 $L \log N$ である．したがって，1 ピアあたりたかだか $L \log N/N$ 件の lookup 履歴しか平均して残らない．各ピアが平均 c 回の検索を行う場合，$L = cN$ と表され，1 ピアに残る履歴の件数は $cN \log N/N = c \log N$ であり，1 ピアが持ちうる履歴情報はまばらであると言える．一方，一般的なオブジェクトの検索を考えた場合，Zipf 分布[6] のように特定少数のオブジェクトに対してアクセスが偏る場合が多い．そのため，その特定少数のオブジェクトに対する検索のみが履歴に残る場合も多くなる．この特定少数のオブジェクトへのアクセスが共通した結果，ピアの推薦を行う場合を抑制するため，今回は平均類似度が 1.0 であるような場合については，推薦を行わないものとした．

また，あるピアに同じピアをくり返し推薦することを抑制するため，過去に推薦を行ったピアの組を登録する集合 supress を用いた．

ピア推薦処理を行った後は Chord の通常の処理と同様に，$targetObject$ がピア n の責任範囲 $(n.predecessor, n]$ 内にあれば問い合わせ処理を終了し，そうでない場合は n 自身の finger テーブルを参照し，$targetObject$ が $[finger[i].start, finger[i+1].start)$ 範囲内にあれば，$finger[i].node$ へ lookup($taregetObject$,$srcPeer$) のルーティングを行う．

以上に示したピア推薦アルゴリズムを擬似コードにより表現したものを図 2 に示す．

4. 評　　価

提案手法によるピアの推薦がどのような性能となるかをシミュレーションにより確認する．

分散ハッシュテーブルを構成するピアの総数を $N = 100$，i 番目のピア $(i = 0, \ldots, 99)$ は $\lfloor i/10 \rfloor$ 番目のグループに属し，同じグループに属するピアは類似した嗜好を持つものとする．lookup のターゲットとなるオブジェクトの総数を $K = 500$ 種類とし j 番目のオブジェクト $(j = 0, \ldots, 499)$ は $\lfloor j/50 \rfloor$ 番目のグループに属するピアから高頻度でアクセスされ，類似した特徴ビットベクトルを持つものとする．

ここで，類似した特徴ビットベクトルの生成は次のように行う．まず，長さが 64 で 16 要素がランダムで 1 である代表ビットベクトルを生成する．この代表ビットベクトルについ

```
n.lookup(targetObject, srcPeer)

  map.initialize(); // intialize mapping: PEER → VAL
  b_t ← features.get(targetObject);
  for each p ∈ accesslog.keyset()
    K ← accesslog.get(p);
    for each k ∈K
      s ← similarity(b_t,features.get(k));
      map.put(p, map.get(p)+s);
    map.put(p, map.get(p)/|K|);

  for each p ∈ map.keyset();
    v ← map.get(p);
    if (v > β and v ≠ 1.0 and (p, srcPeer) ∉ suppress)
      if (p has not been recommended to srcPeer)
          n send a message for recommending p to srcPeer
          suppress.put(p,srcPeer)

  if( targetObject ∈ (n.predecessor, n])
    return n; // found in my responsible range
  for i = 1 to m
    if(taregetObjectId ∈ [finger[i].start, finger[i + 1].start )
      n' = finger[i].node; // forward to n'
      return n'.lookup(taregetObject,srcPeer);
```

図 2 ピア推薦アルゴリズムの擬似コード表現

て，値が 1 である要素を 0 に反転，値が 0 である要素を 1 に反転，という操作を 2 回ランダムに行うことで一つの類似した特徴ビットベクトルを生成する．

各ピアおよび各オブジェクトの識別子はランダムに決定され，分散ハッシュテーブルに分散配置される．

総数 K のオブジェクトのうち j 番目のオブジェクトが検索対象となる確率は，$k = 50 \times \lfloor j \bmod 50 \rfloor + \lfloor j/50 \rfloor$ としたとき，以下に示す Zipf 分布 $P(k, \alpha, K)$ により決定する．

$$P(k, \alpha, K) = (k+1)^{-\alpha} \Big/ \sum_{l=0}^{K-1} (l+1)^{-\alpha}$$

ただし，$\sum_{k=0}^{K} P(k, \alpha, K) = 1.0$，$\alpha$ は Zipf 分布におけるアクセス分布の偏り具合を制御するパラメタであり，本実験においては $\alpha = 1.0$ とする．

ランダムに選択されたピアはまず自身の好みのオブジェクトにアクセスするかどうか否かを確率 γ により判定する．i 番目のピアが自身の好みのオブジェクトにアクセスする場合，オブジェクト集合 $\{j \mid \lfloor i/10 \rfloor = \lfloor j/50 \rfloor\}$ から各オブジェクトが検索対象となる確率の比にもとづいてアクセス対象を選択する．また，i 番目のピアが自身の好みによらずオブジェクトにアクセスする場合，オブジェクトの全体集合から各オブジェクトが検索対象となる確率にもとづいてアクセス対象を選択する．

この検索対象オブジェクトを決定する手順にしたがい検索対象オブジェクトを 100,000 回選んだとき，各オブジェクトに対してどのような頻度でアクセスが行われるかの一例を図 4 に示す．

上記のアクセスパターン生成手順により生成されたシナリオにしたがって各ピアから検索を実行させたシミュレーションを行い，送信されるピア推薦メッセージをすべて記録する．このときの総推薦メッセージ数に対するピアに同じ嗜好を持つグループのピアを推薦するメッセージ数の比を推薦の正確さとし，これを評価指標とする．

まず，好みのオブジェクトにアクセスする確率 γ の変化に伴いピアの推薦の正確さがどのように変化するかの確認を行った．各ピアが推薦メッセージを送信するか否かのしきい値 β は，長さ 64 の 2 つの特徴ビットベクトル間に平均で 4 箇所の差がある場合 $(64-4)/64 = 0.9375$ とし，$\gamma = 0.5, 0, 6, 0.7, 0.8, 0.9$ の 5 パターンのそれぞれについて，システム全体でのオブジェクト検索の回数を $L = 10000$ とし，推薦の正確さの評価を行った．

評価結果を図 4 に示す． 図より各ピアが好みの特徴を持つオブジェクトにアクセスする確率が高いほど推薦が正しく行われることが見て取れる．あるピア P が好み以外のオブジェ

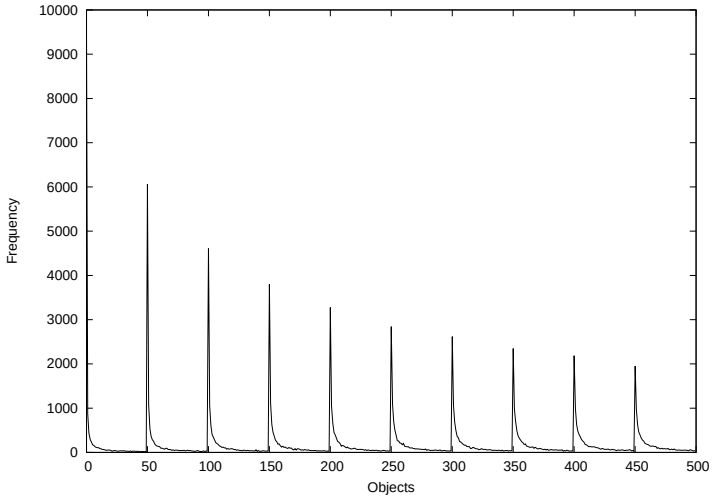


図 3 各オブジェクトに対するアクセス頻度の例

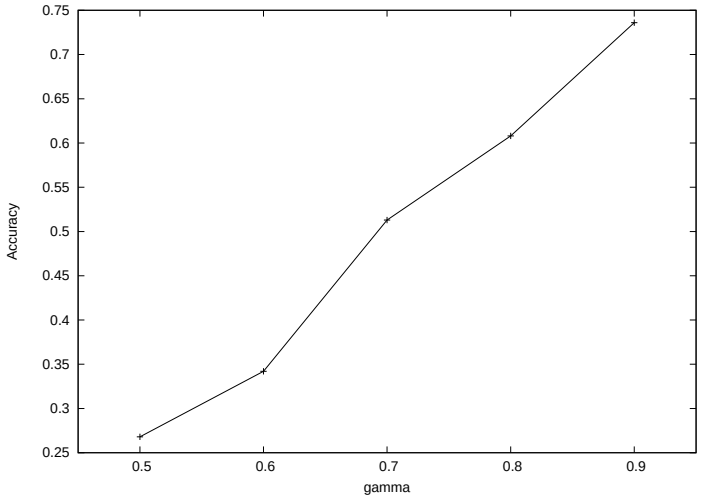


図 4 好みのオブジェクトにアクセスする確率のピア推薦の正確さに対する影響

表 1 ピア推薦を行うしきい値と推薦メッセージ数の関係

β	57/64	58/64	59/64	60/64	61/64	62/64
#recommends	21184.3	10590.0	9683.7	3582.7	432.7	7.7

クト O にアクセスし，その O を好みとするようなグループ G が存在する場合，提案した推薦手法においてはグループ G に属するピアを推薦するメッセージがピア P に送信される．特に特定少数のオブジェクトが人気オブジェクトとなり，所属グループに限らずどのピアもその特定オブジェクトにアクセスする確率が高い今回のようなアクセスパターンの場合には，このオブジェクトへのアクセスが誤った推薦を引き起こすことになる．推薦の正確さを向上させるためには，全体のピアと共通にアクセスされるオブジェクトについては推薦に対する重みを減らすというような考えをもとにピア推薦を行う必要がある．一方，VoD などに代表される有料サービスのバックエンドとして P2P システムを考える場合，好み以外の動画にお金を払ってアクセスする傾向はあまり高くなく，アクセスは好みの特徴を持つオブジェクトに対してのみ行われることが考えられる．その際には比較的良い推薦が行われる可能性がある．

次に，各ピアが推薦メッセージを送信するか否かを決めるしきい値 β を変化させた時の振る舞いを確認した．特徴ビットベクトル間に平均して $7, 6, \ldots, 2$ 箇所の差異がある $\beta = 57/64, 58/64, \ldots, 62/64$ のそれぞれのしきい値について，システム全体でのオブジェクト検索の回数を $L = 10000$ とし，推薦の正確さの比較を行った．

しきい値を変化させた際の推薦の正確さの評価結果を図 4 に，各しきい値ごとに送信された推薦メッセージの数を表 1 に示す．図および表において，しきい値を高い値にするにつれて推薦の正確さは短調に上昇することが確認できるが，一方で推薦メッセージ数は激減することが見て取れる．特に $\beta = 62/64$ の場合においては，1 ピアあたり平均 100 回のオブジェクトの検索を行なう間にシステム全体に流れる推薦メッセージ数は 7.7 であり，推薦メッセージを送ることも受け取ることもほぼ無いと言える．今回の手法においては lookup 処理をトリガーにして推薦メッセージを送るか否かの判定を行っているため，推薦メッセージを送る場合には次のピアへの lookup の転送または lookup の始点のピアへの返信にこの推薦メッセージを便乗させることが可能である．したがってメッセージ数が大幅に少ないことに大きな利点は無いので，しきい値を極端に大きくする必要は無いと考えられる．

最後に，システム全体でのオブジェクト検索の回数の推薦の正確さに対する影響を評価した．各ピアが推薦メッセージの送信を行うか否かを決定するしきい値 β を $60/64 = 0.9375$，各ピアが好みのオブジェクトにアクセスする確率 γ を 0.5, 0.7 とし，システム

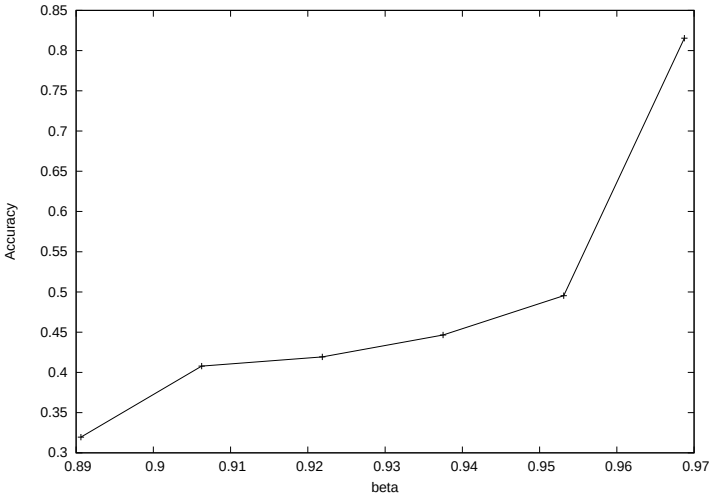


図 5 ピア推薦を行うしきい値のピアの正確さに対する影響

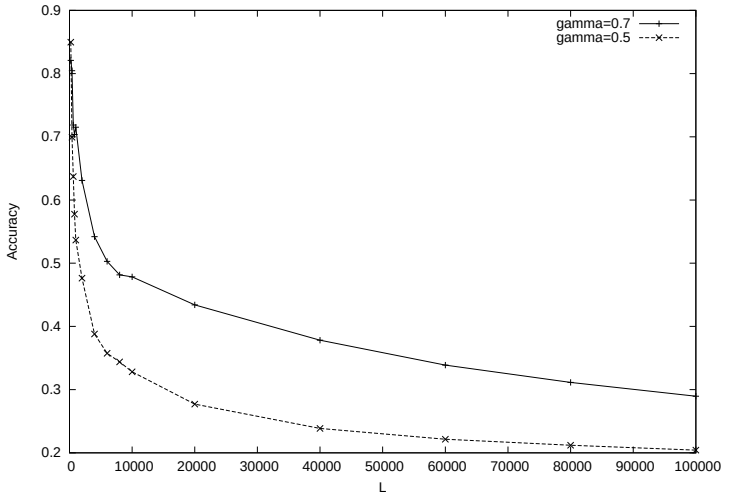


図 6 総 lookup 回数のピア推薦の正確さに対する影響

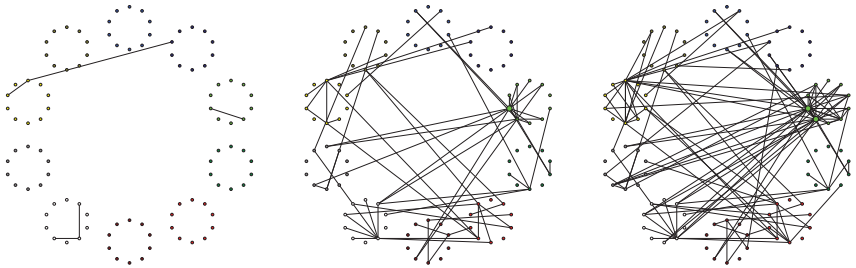
全体でのオブジェクト検索の回数 L を 200,400,600,800,1000,2000,4000,6000,8000,10000, 20000,40000,60000,80000,100000 の 15 通りに変化させ，ピア推薦の正確さを評価した．

評価結果を図 4 に示す．図より，L が増加するにつれて推薦の正確さは著しく低下し，L の増加は推薦の正確さの向上には寄与しないことがわかる．このことから，各ピアのアクセス履歴は大量に保持する必要は無いと言える．しかし，ピア推薦を行うか否かのしきい値 β の影響実験で触れたのと同様に，推薦の正確さが高い L の値が比較的小さい期間においては，推薦メッセージ数が非常に少ない．推薦の正確さと推薦メッセージ数の適切な関係を推定するには，システムを構成するピア総数に対し，どの程度の履歴データを蓄積すべきかという問題について検討する必要がある．ピュア P2P システムにおいては厳密なピア総数を知ることは困難であるが，文献[2] に示されたランダムサンプリングによる手法によりおおよその数の推定は可能であると考えられる．

$\gamma = 0.7$ とし，L が $200, 600, 1000$ と増加するにつれて各ピアがどのようなピア推薦を受けるかを無向グラフにより可視化したものを図 4 に示す．図中小さい環状に並んでいるのが同一の嗜好を持つピアのグループであり，ピア p_1 から p_2 の間にあるエッジ (p_1, p_2) がピア p_1 がピア p_2 の推薦を受けたことを示している．可視化結果からも L が小さい段階では正しい推薦は行われているもののエッジの数が非常に少なく，これをもとにして同じ好みを持つピアのグループを構成することが困難であることがわかる．ピアのグループを組むためには，誤りを含む推薦メッセージを受けることを前提とし，グループを組む前にその推薦が妥当であるかどうかの検証を行う，またはグループを組んだ後に性能の向上が見られない場合にはグループを離脱するといった機構が必要になる．

5. おわりに

本稿において，分散ハッシュテーブルにおける lookup の履歴情報をもとにピア推薦を行う手法についての検討を行った．今回，分散ハッシュテーブルの代表例の一つである Chord に主に履歴を記録するデータ構造を追加し，あるピアが検索の対象としたオブジェクトと履歴のなかで他のピアが検索の対象としたオブジェクトの平均類似度をもとにピアを推薦する手法を述べた．また，この手法に関して，比較的少ないピア数でのシミュレーション実験を通じ，ピアが嗜好にもとづいてアクセスする場合の頻度，推薦を行うか否かを決定するしきい値，履歴サイズの大きさについてピア推薦の正確さがどのような傾向を示すのかをあきらかにし，性能向上のための検討を行った．今後の課題としては，ピア総数をより多くした場



(a) $L = 200$ (b) $L = 600$ (c) $L = 1000$

図 7 推薦状況の可視化

合の振る舞いについての確認，ピアの持つ履歴により高度な統計処理を行うことによる推薦の正確さの向上，ピア推薦をもとにしたグループの構成とグループからの離脱の手法の検討が挙げられる．

参 考 文 献

1) Balakrishnan, H., Kaashoek, M. F., Karger, D., Morris, R. and Stoica, I.: Looking up data in P2P systems, *Commun. ACM*, Vol. 46, pp. 43–48 (online), DOI:http://doi.acm.org/10.1145/606272.606299 (2003).
2) Bharambe, A. R., Agrawal, M. and Seshan, S.: Mercury: supporting scalable multi-attribute range queries, *Proceedings of the 2004 conference on Applications, technologies, architectures, and protocols for computer communications*, SIGCOMM '04, New York, NY, USA, ACM, pp. 353–366 (online), DOI:http://doi.acm.org/10.1145/1015467.1015507 (2004).
3) Iwamaru, A., Itokawa, T., Kitasuka, T. and Aritsugi, M.: Introducing Group Participation Support into P2P Web Caching Systems, *Proceedings of the 2009 International Conference on Advanced Information Networking and Applications*, Washington, DC, USA, IEEE Computer Society, pp. 868–875 (online), DOI:10.1109/AINA.2009.109 (2009).
4) Ratnasamy, S., Francis, P., Handley, M., Karp, R. and Shenker, S.: A scalable content-addressable network, *Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications*, SIGCOMM '01, New York, NY, USA, ACM, pp. 161–172 (online), DOI:http://doi.acm.org/10.1145/383059.383072 (2001).
5) Stoica, I., Morris, R., Liben-Nowell, D., Karger, D.R., Kaashoek, M.F., Dabek, F. and Balakrishnan, H.: Chord: a scalable peer-to-peer lookup protocol for internet applications, *IEEE/ACM Trans. Netw.*, Vol. 11, pp. 17–32 (online), DOI:http://dx.doi.org/10.1109/TNET.2002.808407 (2003).
6) Zipf, G.K.: *Human behavior and the principle of least effort; an introduction to human ecology*, Addison-Wesley Press, Cambridge, Mass., (1949).