P2P 情報検索における単語の重みに基づいたデータ配置手法

倉沢 央[†] 若木裕美[†,☆] 正田備也[††,☆☆]
高須淳宏[††] 安達 淳[††]

Peer-to-Peer(P2P) ネットワークを用いた情報検索では，低コストでありながら負荷分散や高いスケーラビリティが簡単に実現可能である．しかしながら既存の P2P ネットワークを用いた情報検索手法のデータ配置法は，全ての文書のデータが単語に対して均等な重み付けに割り当てられている．つまり，既存の手法では個々の文書のデータは内容に無関係に配置されているため，ユーザは問い合わせに対する適合度の大小に関わらず同じだけの手間をかけて文書を取得しなければならない．そこで我々は，P2P 情報検索における索引とデータの分散配置手法，Concordia を提案する．文書のデータを単語の重みに基づいて配置することで，問い合わせとの適合度の高い文書ほど収集を容易にした．さらに，文書と関係する単語のハッシュ値に該当するノードすべてに，(k, n) 閾値法でエンコードした分散情報を単語の重みに基づいて配置することで，ノードの突如の離脱が起こりうる状況でも安定した文書の収集を実現した．

Data Distribution Based on Term Weight for P2P Information Retrieval

Hisashi Kurasawa ,[†] Hiromi Wakaki ,[†,☆]
Tomonari Masada ,[††,☆☆] Atsuhiro Takasu [††] and Jun Adachi[††]

Many Peer-to-Peer information retrieval systems have already been proposed that can retrieve documents relevant to a query. Since documents are allocated to peers regardless of the query, a user needs to connect many peers to gather the relevant documents. We propose a scheme that uses a new distributed index and data allocation, which is called Concordia, for P2P information retrieval. Concordia uses a node to allocate the binary data of a document based on the weight of each term in the document to efficiently assemble all the documents relevant to a query from the P2P Network. Moreover, the node encodes the binary data of a document with a (k, n) threshold scheme, and Concordia produces an efficient redundancy for counteracting node failures.

1. はじめに

インターネットの登場により人類の生産する情報は爆発的に増加し続けている[12]．これら大量の情報を効率良く扱い，最適な情報を得るものとして，検索エンジンが大きな役割を担っている．現在主流の検索システムは，Google[3] に代表される集中型である．中規模以上の検索システムの構築，運用には，負荷分散やスケーラビリティなどについての専門的な知識と，高いコストが必要となることが知られている．これに対して，低コストでありながら負荷分散や高いスケーラビリティを簡単に実現できるものとして，Peer-to-Peer(P2P) ネットワークを用いた検索システムが近年注目をあびている．

情報検索システムは，ユーザからの問い合わせに適合する文書を探しだし，これら文書を取得したいという要求を満たすためのシステムである．P2P 情報検索では，P2P 環境において問い合わせに適合する文書をいかに効率良く探し出し，そして収集するかが重要な課題である．情報検索における問い合わせは複数の単語から成る．情報検索システムでは問い合わせと蓄積されている文書間の適合度を計算し，適合度の大きい文書をユーザに提示する．問い合わせとの適合度の計算には一般的に，文書におけるある単語の出現頻度を示す tf と，文書コレクション全体におけるある単語を含む文書の出現頻度を示す df が用いられる．P2P

† 東京大学
The University of Tokyo
☆ 現在，東芝 研究開発センター
Presently with Corporate Research & Development Center, Toshiba Corporation
☆☆ 現在，長崎大学
Presently with Nagasaki University
†† 国立情報学研究所
National Institute of Informatics

表 1 Comparison between Concordia and Related Researches

	Indexing	Data Allocation	Search Target	Redundancy
Concordia	DHT	the nodes related to data index	relevant documents	(k, n) threshold scheme
PlanetP	Gossip	publisher	relevant peers	N/A
MINERVA	DHT	publisher	relevant peers	N/A
Freenet	N/A	the nodes around the hash value of the query	relevant documents	making replicas based on needs

のような分散環境において効率的な検索を行うには，これら出現頻度の情報を参照できることが望ましい．しかしながら，P2P 環境では文書のデータは分散してノードに保持されるので，簡単には参照できない．そこで近年，ネットワークにおいて共有されている文書全体を把握する目的で索引の作成を行う手法が提案されている[6),9)]．だが，これら P2P 情報検索の既存手法では出現頻度の情報の利用は限られている．ユーザは問い合わせに対する適合度の大小に関わらず，索引を参照し，文書のオリジナルデータを保持するノードかその複製データを保持するノードから収集する必要がある．そこで，出現頻度の情報を索引から参照でき，さらにトラフィックを抑えた効率の良い収集が可能な P2P 情報検索が必要である．

このために，筆者らは P2P 情報検索における索引とデータの配置手法，Concordia の研究を行ってきた[21)]．Concordia は文書における各単語の重みに基づいて文書のデータをノードに配置する手法である．本稿では Concordia における 2 つのデータ配置方式を提案する．Concordia-1 では，文書における重みの大きな上位 n 単語のハッシュ値に該当するノードに文書のデータの複製を配置する．この手法により，問い合わせとの適合度が高い文書ほど効率的に集めることができるようになる．さらに，Concordia-2 では，文書のデータの複製の代わりに (k, n) 閾値法[19)] を用いて n 個の分散情報にエンコードし，単語の重みに相関した数の分散情報を単語のハッシュ値に該当するノードに配置する．この分割法を用いると，文書は任意の k 個の分散情報から復元できる．つまり，仮に $(n - k)$ 個の分散情報をノードの突如の離脱によって収集できなくても，ユーザはその文書を得ることができる．これにより，ネットワークで扱うデータの総量を抑えながら効率の良い冗長化を実現する．文書コレクションを用いた実験から，Concordia は文書における単語の重みに基づいた配置をすることで，問い合わせに適合する文書ほど容易に収集可能であることを確認できた．さらに，複製を配置する Concordia-1 よりも (k, n) 閾値法を用いる Concordia-2 のほうがファイルの損失確率を抑えることができることも確認できた．提案手法は以下のようにまとめられる．

- Concordia は，P2P 環境において単語と文書の出現頻度を参照できる索引を作成する．索引を用いることで，問い合わせと文書間の適合度を計算することが可能となり，集中型に近い検索を実現している．
- Concordia は，問い合わせ内容に対応する索引を保持するノードに，問い合わせとの適合度の高いファイルを配置する．この配置法により，検索過程で拡散する問い合わせ数が少なくすみ，収集の効率を向上している．

本稿では，まず第 2 章で既存の P2P 情報検索手法の問題点を明らかにする．第 2 章で述べた問題に対し，第 3 章にて P2P 情報検索における索引とデータの分散配置手法について述べる．第 4 章で本手法の評価を行い，第 5 章で考察を行う．

2. 関 連 研 究

P2P 情報検索の関連研究について，索引の作成法，データの配置法，検索対象，そして冗長化という観点から紹介する．

P2P ネットワークを使った情報検索は，問い合わせを近隣ノードへ転送することで欲しい文書を探索する手法が提案され，様々なファイル共有ソフトで実用化されている．Gnutella では，隣接ノードへ検索問い合わせをフラッディングする幅優先探索を行う[2)]．文書のデータはそのオリジナルデータを持つノードとそれを検索して取得したノードに配置される．また，Freenet では，問い合わせのハッシュ値に近いノードへ問い合わせを転送する深さ優先探索を行う[8)]．文書のデータはそのオリジナルデータを保持するノードからそれを検索したノードまでの経路上に配置される．これらの手法は柔軟な問い合わせに対応できるという利点を持つ．しかし，ネットワークにおいて共有されている文書全体の把握が難しいという欠点を持つ．つまり，問い合わせを転送するのみでは，検索によって得られた文書が他の文書よりも問い合わせと適合しているかを正しく判断できない．仮にネットワークにおいて共有されている文書全体を把握しようとするとな

るべく多くのノードへ問い合わせを転送することになり，大量のトラフィックを引き起こしてしまう．

そこで，このトラフィックの問題を解決するため，集中型の検索システムと同じく，文書の索引の作成を行う手法が近年提案されている．索引を用いた P2P 情報検索は，索引を参照することによりある程度絞り込んだノードに対して問い合わせを投げることができ，効率的な検索が可能となる．Rutgers University の PlanetP は Unstructured 型の P2P 情報検索である[9]．PlanetP では，ノードは Gossip アルゴリズムを用いて索引をフラッディングして，ノードの持つ文書コレクションについての情報を共有する．それぞれのノードはローカルに保持する文書コレクションに含まれる単語すべてを関連づけた Bloom Filter[7] を作る．そのようにして作られた Bloom Fileter をまとめたものが索引となる．文書のデータはオリジナルを持つノードと，それを検索して取得したノードに配置される．一方，Planck-Institut fuer Informatik の MINERVA は Structured 型の P2P 情報検索である[6]．MINERVA では，ノードは Distributed Hash Table(DHT) を利用した構造的な索引を作成する．ノードが担当する単語は DHT によって決められる．それぞれのノードはローカルに保持する文書コレクションの概要を作成し，その概要を文書コレクションに含まれる単語を担当するノードに送信する．その結果，各ノードは，担当する単語とノードのアドレスの索引を保持することになる．さらに MINERVA では，ノードの持つ文書における単語の占める割合を考慮した評価を算出し，接続するノードをさらに絞り込む設計になっている．ノードの選択手法として，各ノードの持つ文書コレクションの重複度合いを考慮して，収集する文書の再現率を高める研究もされている[5]．文書のデータは，PlanetP と同様に，オリジナルを持つノードと，それを検索して取得したノードに配置される．

文書の収集という観点では，Freenet は問い合わせに基づいた文書のデータの配置法であり，問い合わせに適合する文書の収集が容易であると言える．Freenet では問い合わせの経路上のノードに問い合わせを満たす文書のデータの複製が配置されるため，文書のデータはその文書を要求する問い合わせのハッシュ値に近いノードに配置されやすい．しかしながら，出現頻度の低い問い合わせに対しては，Freenet のノードはランダムに文書のデータを配置するシステムと同様の手間をかけて文書を収集することになる．一方，索引を作成する研究ではいずれも，文書のデータの配置法は全ての文書が単語と無関係なノードに割り当てられている．つまり，索引を参照することでノードを絞り込むことはできても，ユーザは問い合わせに対する適合度の大小に関わらず同じだけの手間をかけて文書を取得しなければならない．P2P 情報検索は，集中型の検索システムと同様に共有する文書の索引を作成することで問い合わせに適合する文書を検索でき，さらに問い合わせに基づいた文書のデータの配置をすることで効率の良い収集が可能であることが望ましい．

以上を踏まえて，P2P 情報検索における新たな索引とデータの配置手法を提案する．本提案手法を用いることで，ノードは集中型検索システムのように文書を検索でき，問い合わせの内容の出現頻度に関わらず文書を効率よく収集できる．さらに，筆者らはノードの突如の離脱を想定した共有する文書のデータの冗長化が必要であると考える．筆者らが提案する Concordia と関連研究とを比較したものが表 1 である．

3. Concordia のアーキテクチャ

Concordia の機能的な特徴は次のようにまとめられる．

- DHT を利用して文書ごとに単語の索引を作成することで，ユーザは問い合わせとの適合度を文書ごとに算出可能である．
- 文書に含まれる単語の重みに基づいてノードに文書のデータの複製を配置することで，問い合わせに適合する文書を効率よく収集可能である．
- 文書のデータを (k, n) 閾値法を用いてエンコードすることで，ノードの突如の離脱を想定した効率の良い冗長化が可能である．(Concordia-2)

3.1 Concordia の概要

Concordia は集中型検索システムに近い文書の検索と順位付けができ，さらに問い合わせに適合する文書を効率良く収集できる手法である．Concordia は文書のデータをその文書に含まれる単語それぞれの重みに基づいてノードに配置する．各ノードは担当する単語の索引とその単語に対して重要度の高い文書のデータを管理する．

3.1.1 Concordia-1

Concordia-1 では，ノードはローカルに保持する文書それぞれについて単語の重みを求めて，DHT 上に索引を作成する (第 3.2 章，第 3.5 章)．また，ノードは文書ごとに文書における重要な単語 (式 (2)) を文書を参照する索引に登録する．それと同時に，ノードは文書における重みの大きな上位 n 単語のハッシュ値に該当するノードに文書のデータの複製を配置する．Concordia-1 における索引とデータの配置法を示した

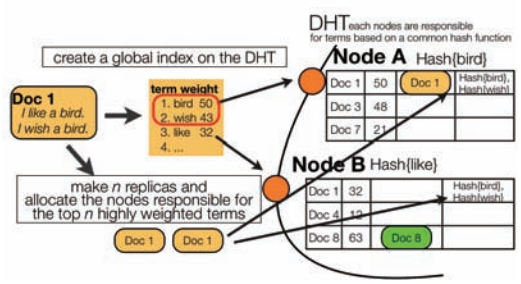

図 1 Indexing and Data Allocation (Concordia-1)

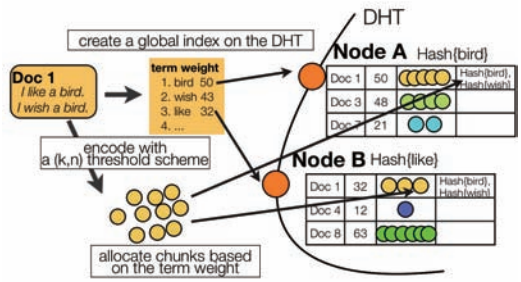

図 2 Indexing and Data Allocation (Concordia-2)

ものが図 1 である．

ユーザが問い合わせに適合する文書を発見・収集するには，まず問い合わせから単語を抽出し，その単語の索引を保持するノードに接続し，索引を参照する．複数の索引を参照することで問い合わせとの適合度を求め，適合度の高い文書を収集する．接続したノードに文書が存在しない場合は，索引に記述されたその文書における重要な単語を参照し，その単語を担当するノードから文書を収集する．

3.1.2 Concordia-2

Concordia-2 では，ノードはローカルに保持する文書それぞれについて単語の重みを求めて，DHT 上に索引を作成する (第 3.2 章，第 3.5 章)．また，ノードは文書ごとに文書における重要な単語 (式 (2)) を文書を参照する索引に登録する，それと同時に，文書のデータを (k, n) 閾値法[19] を用いて n 個の分散情報にエンコードする．(k, n) 閾値法を用いることで，任意の k 個の分散情報から元の文書を復元可能となる．そして，ノードは文書における単語の重みに基づいた数の分散情報を単語のハッシュ値に該当するノードに配置する (第 3.4 章)．つまり，文書における重要な単語を担当するノードほど，その文書の分散情報をより多く保持することになる．この配置法にすることで，たとえ文書の分散情報を持ついくつかのノードが離脱しても，その文書の復元は可能である．Concordia-2 における索引とデータの配置法を示したものが図 2 である．

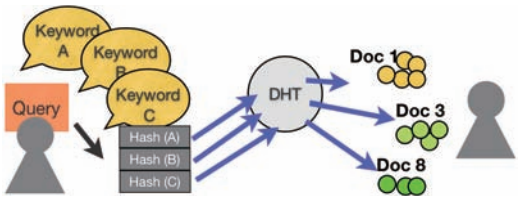

図 3 Retrieving and Gathering (Concordia-2)

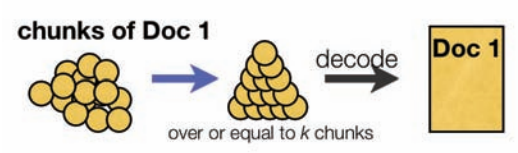

図 4 Decoding Documents (Concordia-2)

ユーザが問い合わせに適合する文書を発見・収集するには，まず問い合わせから単語を抽出し，その単語の索引を保持するノードに接続し，索引を参照する．これらノードは，担当する単語に関係する文書の分散情報を保持している．保持する分散情報の量は文書における単語の重みに相関する．ユーザは複数の索引を参照することで問い合わせとの適合度を求め，適合度の高い文書の分散情報を収集する．Concordia-2 における検索と収集法を示したものが図 3 である．仮にユーザが適合率を重視するときは，問い合わせとの適合度の高い文書の分散情報を集めることになる．一方，仮にユーザが再現率を重視するときは，ノードが保持する分散情報をすべて集めることになる

問い合わせに適合する文書の分散情報を発見・収集した後，集めた分散情報のうち閾値を上回っている文書はデコードする．分散情報が閾値を満たさない文書は，索引を参照してその文書における重要な単語を調べ，そのハッシュ値を管理するノードに接続して足りない分散情報を新たに収集する．Concordia-2 における文書の復元を示したものが図 4 である．

3.2 P2P 環境での単語の重み付け

Concordia を用いた文書の検索と収集をより効率の良いものにするうえで，単語の重み付けは最も重要な要素である．情報検索の代表的な手法に，ベクトル空間モデル (Vector Space Model) と確率モデル (Probabilistic Model) がある．

本研究では，単語の重み付けに Okapi で採用された BM25[15] と呼ばれる確率モデルの式を元にした式を用いた[10]．文書 d における単語 t の重みは次のよ

うに定義する．

$$w(t,d) = \log \frac{N}{df} \cdot \frac{(k+1)\cdot tf}{k\{(1-\alpha)+\alpha\cdot\frac{dl}{avdl}\}+tf} \quad (1)$$

$w(t,d)$ は文書 d における単語 t の重み，k と α は定数，N は文書コレクションに含まれる文書の総数，tf は d における t の出現頻度，df は全文書における t を含む文書数，dl は d の長さ，$avdl$ は文書の長さの平均値を表す．

問い合わせ Q との適合度は以下の式を用いる．

$$R(Q,d) = \sum_{t\in Q} w(t,d). \quad (2)$$

P2P 環境にて式 (2) で示した単語の重みを計算する場合，文書の総数 N と文書長の平均値 $avdl$，全文書におけるある単語を含む文書数 df，そして文書における単語の出現頻度 tf が必要になる．P2P のような自律分散環境において，文書はネットワーク全体に分散して配置されているため N，$avdl$，df の 3 つの値を取得することが難しい．そこで，我々は既存研究[6)18)] と同じように DHT を用いて構造的に単語の索引を作成し，DHT 上に配置された情報を元に df の値を得ることにした．各ノードは保持する文書すべてについて，文書に含まれる単語を担当するノードに接続し，索引に文書の情報を登録する．その結果，すべてのノードは担当する単語についての索引を管理する．ユーザは索引を参照することである単語を含む文書数，つまり df を得ることができる．

文書の総数は動的に変化するため，筆者らは文書の総数 N と文書の長さの平均値 $avdl$ の値の算出は難しいと考える．そこで，本研究では簡略化し，適切と思われる定数を設定することにした．

3.3 (k,n) 閾値法を用いた文書の分割法

Concordia-2 の文書のデータを (k,n) 閾値法で n 個の分散情報にエンコードする．(k,n) 閾値法を用いることで，文書は任意の k 個の分散情報から復元できる．つまり，仮に $(n-k)$ 個の分散情報をノードの突如の離脱によって収集できなくても，ユーザはその文書を得ることができる．(k,n) 閾値法は $(k-1)$ 次関数上の n 個の座標を用いたもの[16)] や，k 個の文字列を使った連立方程式を用いたもの[11)] がある．いずれも文書の復元は k 個の連立方程式の解を求める計算となる．

(k,L,n) ランプ型秘密分散法[19)] は，分散対象の情報を $(L+1)$ 個のビット連結として扱い，それを $(k-1)$ 次関数の係数とする手法である．個々の分散情報のサイズは文書のデータサイズの $\frac{1}{L}$ となる．式で表すと以下のようになる．

$$S = S_0||S_1||S_2||\cdots||S_L \quad (3)$$

$$y = S_0 + S_1 x + \cdots + S_L x^L + r_1 x^{L+1} + \cdots + r_{n-L} x^{k-1} \mod p, \quad (4)$$

S は文書のデータ，|| はビット連結演算子，p は $p > max(S_i)(0 \le i \le L)$ を満たす素数を示す．分散符号化された分散情報はこの $k-1$ 次関数上の (x,y) の座標情報 n 個となる．個々の分散情報のサイズ m は $m \ge \frac{S}{L}$ となり，分散情報のサイズの総量は $\frac{n\cdot S}{k}$ となる．

本研究ではこの (k,L,n) ランプ型秘密分散法を用いた．L の値はデータサイズを小さくするため $L-1=k$ とした．

3.4 単語の重みに基づいたデータ配置法

P2P 情報検索では，問い合わせに適合する文書を探すだけでなく，効率よく適合する文書を収集できることが重要である．問い合わせに対応する索引を保持するノードが，問い合わせとの適合度の高い文書を保持していると，ユーザが検索過程で拡散する問い合わせ数が少なくすみ，検索と収集の効率が良いと言える．しかし，問い合わせを投げたときにアクセスする可能性の高いノードすべてに文書のデータの複製を配置すると，ネットワーク全体で扱うデータの総量が大きくなってしまう．我々は収集効率を維持しながらデータの総量を抑えることを考えた．

文書の問い合わせとの適合度は，式 (2) で述べたように，問い合わせに含まれる単語の文書における重みの和を用いる．それゆえ，文書における単語の重みに基づいて，データを単語の索引を管理するノードに配置することで，問い合わせとの適合度の高い文書ほど収集が容易になる．

本研究では以上の理由から，Concordia-1 では，文書における重みの大きな上位 n 単語のハッシュ値に該当するノードに文書のデータの複製を配置する．また，Concordia-2 では，文書と関係する単語のハッシュ値に該当するノードすべてに，(第 3.3 章) で述べた分散情報を単語の重みに基づいて配置することにした．つまり，文書における重みの大きい単語ほど多くの文書の分散情報が割り当てられるように配分する．

3.5 DHT を用いたノードとファイルの管理

DHT はノードのアドレスとデータを共通のハッシュで管理するシステムである．ノードとデータは共通のハッシュによって結びつけられており，各ノードは自分のアドレスとハッシュ値の近いデータを管理する．その結果 DHT はスケーラビリティ，耐障害性を備えた自律分散システムになっている．代表的な例として，円構造の Chord[17)]，木構造の Tapestry[20)]，多次元空

間の CAN[14]，XOR を用いる Kademlia[13] などが挙げられる．

Concordia では DHT を用いて文書の索引の作成と管理を行う．個々のノードは自分のアドレスとハッシュ値の近い単語を担当する．ノードは担当する単語を含む文書の情報を受け取り，索引に追加していく．本研究では DHT に Kademlia を採用した．Kademlia を用いることで，ノードは問い合わせとして投げたハッシュ値に近い k 個のノードのアドレスを $O(\log N)$ で取得できる．

4. 評　　価

Concordia は，既存研究と比べて，P2P 情報検索において問い合わせとの適合度の高い文書ほど収集を容易にすることを目指している．さらに，Concordia-2 では (k, n) 閾値法を用いることで，突如のノードの離脱に耐えうるデータの冗長化を備えていることが特徴である．

4.1 インデックスの作成

関連研究[6),9)] では，ノードの持つ文書コレクションを単位として，文書コレクションに含まれる単語を索引に登録している．この手法では，同一ノードの文書コレクションにおける文書を索引で区別することができない．一方，Concordia では文書を単位として，文書に含まれる単語を索引に登録している．つまり，索引は同一ノードが保持する文書を区別可能である．

MINERVA[6] においてノードの持つ文書コレクション C_{all} を索引に登録するのに必要な時間 T_1 を考える．単語 t の重みの計算に必要な時間を $f(t)$ としたとき，T_1 は，

$$T_1 = \sum_{t \in C_{all}} f(t). \tag{5}$$

文書コレクションに含まれる単語数を N_{all} としたとき，MINERVA における索引への登録数は，

$$N_{all}. \tag{6}$$

一方，Concordia においてノードの持つ文書を索引に登録するのに必要な時間 T_2 は，

$$T_2 = \sum_{i=1}^{N_{doc}} \left(\sum_{t \in C_{D_i}} f(t) \right), \tag{7}$$

C_{D_i} は文書 D_i に含まれる単語集合，N_{doc} は文書に含まれる単語数を表す．Concordia における索引への登録数は，

$$\sum_{i=1}^{N_{doc}} N_{D_i}, \tag{8}$$

N_{D_i} は C_{D_i} に含まれる単語数を示す．

インデックスの作成時間は関連研究ではノードの数に依存する．一方，Concordia では共有する文書の数に依存する．Concordia は既存手法と比較して，ノードが保持するファイル間で単語の重複が多いほどインデックスの作成時間と登録数が増加する．

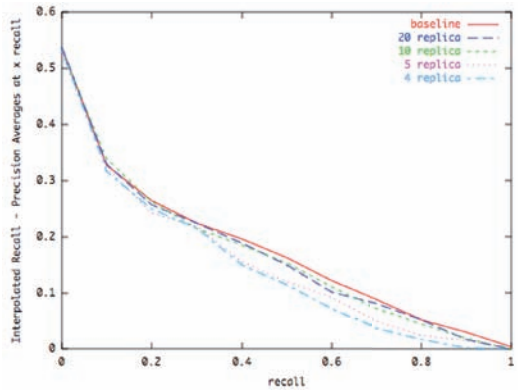

図 5 Interpolated Recall - Precision Average (Concordia-1)

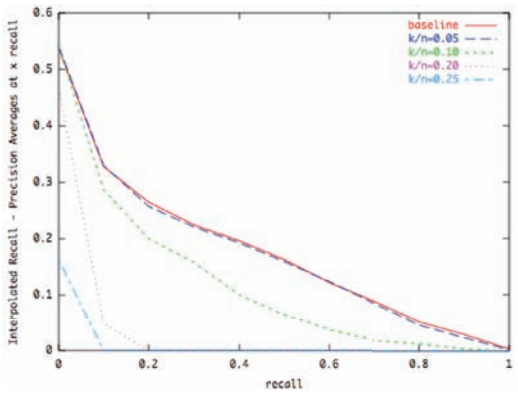

図 6 Interpolated Recall - Precision Average (Concordia-2)

4.2 索引への問い合わせの接続のみで収集できる文書

Concordia では，まずはじめに問い合わせに含まれる単語のハッシュ値に対応するノードからデータを収集する．収集するデータは，Concordia-1 では文書のデータの複製であり，Concordia-2 では文書の分散情報となる．この時点で接続するノード数は，問い合わせに含まれる単語の数と一致する．この新たに文書や分散情報を収集する以前の段階で，どの程度の適合率と再現率を満たすことができるか，実験を行った．

実験に使う文書コレクションとして，TREC[1] の Text Research Collection Volume 3 を用いた．the San Jose Mercury News (1991), the Associated Press (1990), U.S. Patents (1983-1991), そして In-

formation from the Computer Select disks (1991, 1992) copyrighted by Ziff-Davis の 4 種類からなる合計 33.6 万の文書データである．英単語のステミングには Porter stemmer[4] を用いた．ストップワードの除去を行い，ステミングをした後，式 (2) を用いて単語の重みの算出をした．重み付けの式のパラメータは今回は k は 100，α は 0.5 を用いた．TREC-2 ad hoc & TREC-3 routing topics のトピック 50 件それぞれについて，索引への問い合わせの接続のみで収集できる文書をシミュレーションした．ネットワークにおけるノードは，簡単のため文書コレクションに出現する単語のハッシュ値の数だけ存在するとした．適合率と再現率の評価には TREC Eval を使用した[1]．Concordia-1 の結果が図 5 であり，Concordia-2 の結果が図 6 である．適合度の高い文書をすべて収集済みのときの結果を Baseline として示している．

どちらの手法も，新たに文書や分散情報を収集することのない時点でも，冗長の度合いが大きいときは高い適合率と再現率を実現できることがわかる．冗長の度合いが小さいときは，必要なデータが不十分で，新たなデータの収集が必要不可欠であることもわかる．同じだけ文書を冗長化させたときの Concordia-1 と Concordia-2 を比べると，Concordia-1 のほうが追加の収集がなくても検索性能が良いことがわかる．

4.3 問い合わせと適合するファイルの収集効率

Concordia を用いることで，問い合わせに適合する文書の収集が容易になるかについて実験を行った．文書の収集の容易さは，ユーザが問い合わせと適合する文書の収集課程で拡散する問い合わせ数で評価した．

実験に使う文書コレクション及び処理は第 4.2 章と同様のものにした．TREC-2 ad hoc & TREC-3 routing topics のトピック 50 件それぞれについて，適合度の順位付けを行い，適合度の上位 n 文書を取得するのに接続するノード数をシミュレーションした．DHT の機能としてノードのアドレスを問い合わせるときに接続するノードの数は，今回の実験の議論の範疇に含まれないので，実験結果から除いた．Concordia-1 と既存手法を比較した結果が図 7 であり，Concordia-2 と既存手法の比較の結果が図 8 である．既存手法は Baseline として示している．

Concordia が文書の収集の過程で接続するノード数は，問い合わせと無関係なノードに配置する手法よりも少ないことがわかる．つまり，Concordia は単語と関連するノードに文書のデータを配置することで，ユーザは問い合わせとの適合度の高い文書の収集を効率良く行えると言える．また，冗長であればあるほど

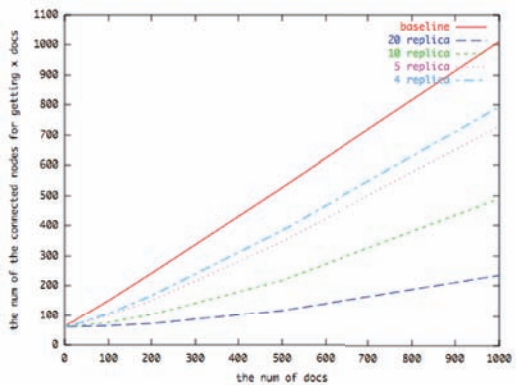

図 7 Number of Connected Nodes for Gathering the Top x Relevant Docs (Concordia-1)

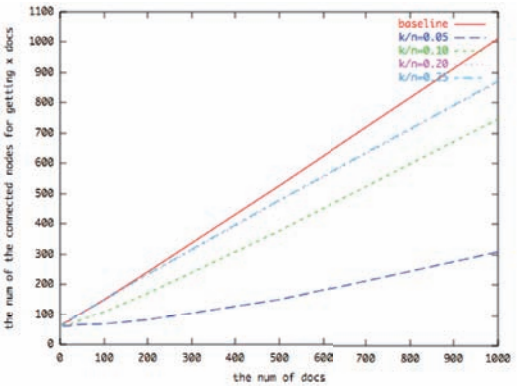

図 8 Number of Connected Nodes for Gathering the Top x Relevant Docs (Concordia-2)

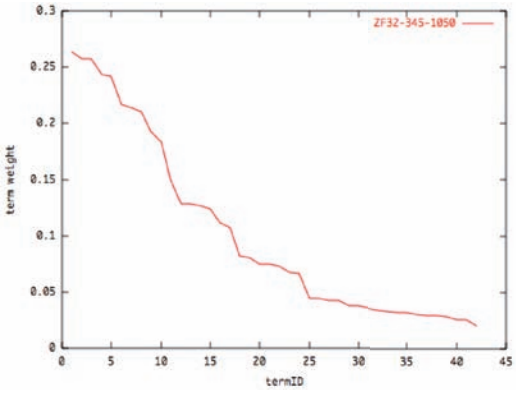

図 9 Term Weight Distribution (ZF32-345-1050)

文書の収集が容易であることもわかる．同じだけ文書を冗長させたときの Concordia-1 と Concordia-2 を比べると，Concordia-1 のほうが効率良く文書を収集できることがわかる．

表 2 Comparison of Redundancy

	Concordia-1	Concordia-2
coding method	replica	(k,n) *threshold scheme*
num of nodes holding data	$\frac{n}{k}$	$\frac{n}{k} \sim n$
file size (piece)	F	$\frac{F}{k}$
file size (total)	$\frac{F \cdot n}{k}$	$\frac{F \cdot n}{k}$
document loss probability	$p^{\frac{n}{k}}$	$\sum_{i=0}^{k-1} {}_nC_i p^{n-i} \cdot (1-p)^i \sim p^{\frac{n}{k}}$

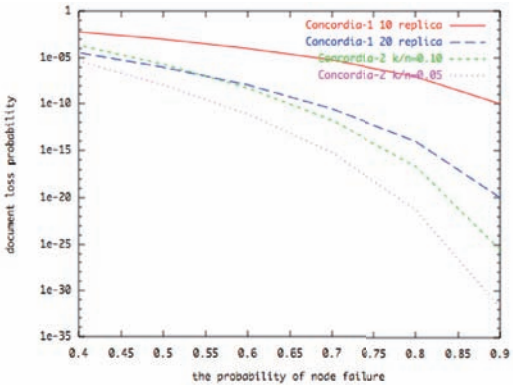

図 10 Document Loss Probability (ZF32-345-1050)

4.4 (k,n) 閾値法による冗長化

Concordia-2 では，単純な複製ではなく，(k,n) 閾値法を用いることにより，単純な文書のデータの複製を配置するよりも，ノードの突如の離脱が起きる環境においても耐えうるシステムを目指している．ノードが頻繁に離脱する状況で文書の収集がどの程度困難になるか，単純な複製を用いる Concordia-1 の冗長化と (k,n) 閾値法を用いる Concordia-2 の冗長化とで比較したものが表 2 である．Concordia-2 における文書のデータの損失確率は分散情報の配分に依存する．

我々は，Concordia における文書のデータの損失確率についてシミュレーションを行った．実験には，TREC[1] の Text Research Collection Volume 3 に含まれる文書番号 ZF32-345-1050 を使用した．単語の重みの分布は図 9 である．Concordia-1 と Concordia-2 において，ノードの離脱率を変化させたときの文書の損失確率を示したものが図 10 である．

冗長化の度合いが同じとき，Concordia-2 のほうが Concordia-1 に比べて文書を安定して取得できることがわかった．つまり，Concordia-2 は (k,n) 閾値法を用いることにより，単純な複製を用いる Concordia-1 よりも効率の良い冗長化が実現できていることがわかる．

5. おわりに

本稿では，P2P 情報検索における索引と文書の分散配置手法，Concordia を提案した．具体的には，文書のデータを単語の重みに基づいて配置することで，問い合わせとの適合度の高い文書ほど収集を容易にする手法である．Concordia-1 では，文書における重みの大きな上位 n 単語のハッシュ値に該当するノードに文書のデータの複製を配置することで，拡散する問い合わせ数を抑えた高い収集効率を実現した．また，Concordia-2 では，文書と関係する単語のハッシュ値に該当するノードすべてに，(k,n) 閾値法でエンコードした分散情報を単語の重みに基づいて配置することで，ノードの突如の離脱が起こりうる状況でも安定した文書の収集を実現した．

現在は，大規模分散環境にて実際に検索の実験を行い，Concordia における文書の検索と収集の応答時間の評価を行うための準備をしている．また，少ない有用な問い合わせに対してさらに効率よく収集可能にすべく，ローカルに持つ情報のみを用いた軽量な問い合わせ拡張などを検討中である．

参 考 文 献

1) *Text REtrieval Conference (TREC)* (http://trec.nist.gov/).
2) *Gnutella* (http://www.gnutella.com/).
3) *Google* (http://www.google.com/).
4) *Porter stemmer* (http://www.tartarus.org/martin/PorterStemmer/).
5) Bender, M., Michel, S., Triantafillou, P., Weikum, G. and Zimmer, C.: Improving Collection Selection with Overlap Awareness in P2P Search Engines, *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR '05)*, pp.67–74 (2005).
6) Bender, M., Michel, S., Triantafillou, P., Weikum, G. and Zimmer, C.: MINERVA: Collaborative P2P Search, *Proceedings of the 31st International Conference on Very Large Data Bases (VLDB '05)* (2005).
7) Bloom, B.H.: Space/Time Trade-offs in Hash Coding with Allowable Errors, *Commn. ACM*, Vol.13, No.7, pp.422–426 (1970).
8) Clarke, I., Sandberg, O., Wiley, B. and Hong,

T.W.: Freenet: A Distributed Anonymous Information Storage and Retrieval System, *Lectured Notes in Computer Science*, Vol.2009, pp. 46–66 (2001).

9) Cuenca-Acuna, F.M., Peery, C., Martin, R.P. and Nguyen, T.D.: PlanetP: Using Gossiping to Build Content Addressable Peer-to-Peer Information Sharing Communities, *Proceedings of the 12th International Symposium on High Performance Distributed Computing (HPDC '03)* (2003).

10) Fang, H., Tao, T. and Zhai, C.: A Formal Study of Information Retrieval Heuristics, *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR '04)*, pp. 49–56 (2004).

11) Gkantsidis, C. and Rodriguez, P.: Network Coding for Large Scale Content Distribution, *Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '05)* (2005).

12) Lyman, P., Varian, H. R., Swearingen, K., Charles, P., Good, N., Jordan, L. L. and Pal, J.: How much information 2003 ?, *http://www.sims.berkeley.edu/research /projects/how-much-info-2003/* (2003).

13) Maymounkov, P. and Mazieres, D.: Kademlia: A Peer-to-Peer Information System Based on the XOR Metric, *Proceedings of IPTPS'02* (2002).

14) 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)*, pp. 161–172 (2001).

15) Robertson, S. E., Walker, S., Jones, S., Beaulieu, M. M.H. and Gatford, M.: Okapi at TREC-3, *Proceedings of TREC-3*, pp.109–126 (1994).

16) Shamir, A.: How to Share a Secret, *Commn. ACM*, Vol.22, No.11 (1979).

17) Stoica, I., Morris, R., Karger, D., Kaashoek, M. F. and Balakrishnan, H.: Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications, *Proceedings of the 2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM '01)*, pp.149–160 (2001).

18) Tang, C. and Dwarkadas, S.: Hybrid Global-Local Indexing for Efficient Peer-to-Peer Information Retrieval, *Proceedings of the Symposium on Networked Systems Design and Implementation (NSDI)* (2004).

19) Yamamoto, H.: On Secret Sharing Systems Using (k,L,n) Threshold Scheme, *IECE Trans*, Vol.J68-A, No.9, pp.945–952 (1985).

20) Zhao, B., Kubiatowicz, J. and Joseph, A.: Tapestry: An Infrastructure for Fault-tolerant Wide-area Location and Routing, Technical Report USB//CSD-01-1141, U. C. Berkeley Technical Report (2001).

21) 倉沢　央，正田備也，高須淳宏，安達　淳：P2P情報検索における索引とファイルの分散配置手法，情報処理学会研究報告, OS-105-21 (2007).