情報処理学会論文誌　データベース　Vol. 1　No. 3　1–10 (Dec. 2008)

P2P 情報検索における単語の頻度情報に基づくデータ配置手法

倉沢　央[†1]　若木裕美[†1,*1]
高須淳宏[†2]　安達淳[†2]

Peer-to-Peer（P2P）ネットワークを用いた情報検索（IR）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の多く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，ユーザは問合せに対する適合度の大小にかか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 IR における検索実行時のコストを削減するためのデータ配置法，Concordia を提案する．Concordia は文書データの配置場所を，検索時に索引参照のために接続するノードと関連づけ，文書における重みの大きな単語の索引を管理するノードに文書データを配置する．文書のデータを単語の重みに基づいて配置することで，クエリとの適合度の高いファイルほど収集を容易にする手法である．この提案手法の効果を評価するため，処理効率を重視した単純複製配置法とノードの頻繁な離脱に対応可能な符号化複製配置法を提案し，性能を実験的に検証した．その結果，適合文書収集時に参照するノード数を削減した高い収集効率を確認した．

Data Allocation Scheme Based on Term Frequency for P2P Information Retrieval

Hisashi Kurasawa,[†1] Hiromi Wakaki,[†1,*1]
Atsuhiro Takasu[†2] and Jun Adachi[†2]

In order to overcome disadvantages of centralized search systems, Peer-to-Peer information retrieval (P2P IR) systems are considered a scalable and cost-effective approach with load balancing capability. Many Peer-to-Peer information retrieval systems that use a global index have already been proposed that can retrieve documents relevant to a query. Since documents are allocated to peers regardless of the query, the system needs to connect many peers to gather the relevant documents. We propose a new data allocation scheme for P2P information retrieval that we call Concordia. Concordia uses a node to allocate 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. To evaluate the efficiency of Concordia, we propose two data allocation methods. One is designed to gather relevant documents efficiently by allocating document replicas. The other is designed to handle frequent node leave by allocating chunks of encoded documents using an erasure code. Experimental results show that Concordia reduces the number of node accesses significantly to gather relevant documents.

1. 序　　論

情報検索のためのアルゴリズムや情報提示手法が数多く提案されているが，実際にユーザに利用されている手法はごくわずかである．その理由として，大規模な検索索引は構築・管理が難しく，検索手法を簡単に実用化する枠組みがないことがあげられる．従来の集中型検索では，大規模な検索索引を構築するための高度な負荷分散技術と，システム増強のための十分な資金が必要である．また，検索エンジンが提供する API を利用しても，検索結果の取得数や利用回数に制限があり，情報提示処理に必要な情報が得られるとは限らない．

一方，ピア・ツー・ピア（P2P）環境で集中型検索のように索引を構築する P2P 情報検索手法（P2P IR）が提案され，近年注目を浴びている．P2P は，耐障害性やアドホック性，スケーラビリティに優れており，大規模システムに向いたアーキテクチャである．これまでの P2P では，分散した文書の管理が難しく，情報検索システムに用いられることはなかった．しかし，文書中の各単語の出現頻度（tf）と文書コレクション全体での各単語の出現頻度（df）を参照できる検索索引が考案され，P2P ネットワーク上での文書検索が容易になった．P2P IR は単語の出現頻度情報を P2P で管理できるため，従来の集中型検索の手法と親和性が高い．既存研究では，分散管理された文書から検索問合せに適合する文書を探す手法が主であり，適合文書を効率良く収集する手法については十分に検討されていなかった．

そこで，我々は，P2P IR において検索索引を用いて適合文書を効率良く取得するためのデータ配置手法である Concordia を提案してきた[12]．Concordia では，文書の重要な語の

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

検索索引を管理するピアに，その文書を配置することによって，P2P ネットワーク上での適合文書取得の効率化を図る．これにより，問合せに適合する文書ほど収集が容易になり，P2P 情報検索の実行時間を削減できる．

本稿では，Concordia のデータ配置法について説明するとともに，実験により適合文書を効率的に収集可能であることを示す．また，Concordia のデータ配置のアイディアに基づいたデータの複製配置法として，単純複製配置法と符号化複製配置法を提案する．単純複製配置法は処理効率を重視した手法であり，符号化複製配置法はノードの頻繁な離脱に対応可能な手法である．実験により，それぞれの特性を示す．

本稿の構成は以下のとおりである．まず 2 章で既存の P2P 情報検索手法の問題点を明らかにする．2 章で述べた問題に対し，3 章で 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．4 章で本手法の評価を行い，5 章で考察を行う．

2. 関 連 研 究

本章では，P2P IR の関連研究を，適合文書の検索方法，データの配置方法という 2 つの観点から述べる．

まず，P2P IR における適合文書の検索方法について述べる．初期の P2P IR では，検索問合せを近隣ノードへ転送することで目的の文書を探索していた[1),7)]．これらの手法では検索の再現率を高めるためには，なるべく多くのノードへ問合せを転送することが必要となり，大量のトラフィックを引き起こしてしまう．そこで，ノードの保持する文書の情報を共有して検索の効率化を図った研究が近年提案されている．

Unstructured P2P では，類似文書を持つノードでクラスタを形成する手法[11),23)]，ノードの持つ文書情報を軽量化して全ノードで共有する手法[8),15)] などがある．

一方，Structured P2P では，文書の単語ベクトルを DHT のハッシュ空間に投影する手法[20)]，単語ごとにノードリスト（term-to-peer 索引）を DHT 上に構築する手法[5),14)]，単語ごとに文書リスト（term-to-document 索引）を DHT 上に構築する手法[16),19),22)] などがある．

これらの P2P IR のうち，term-to-document 索引を構築する手法が最も集中型の転置索引を P2P 環境で再現でき，従来から研究されている検索アルゴリズムを適用しやすい．

次に，P2P IR におけるデータの配置方法について述べる．従来から P2P では，適合データ収集の効率化とロバスト性の確保のため，文書データを冗長にして適切なノードに配置する手法が提案されている．特に Unstructured P2P は，データの配置場所が Structured P2P に比べて制約がないため，データ配置法の研究がさかんである．

Unstructured P2P におけるデータの複製を利用したデータ配置法として，文書をニーズに基づいて冗長にし，それらを必要とするノードが見つけやすい場所に配置した Freenet[7)] があげられる．Freenet は文書への要求度合いを複製生成過程に反映させることで，適合文書の収集効率だけでなく負荷分散にも貢献している．しかし，この手法では出現頻度の低い問合せに対しては，膨大な手間をかけて文書を収集することになる．冗長化を工夫した手法としては，Network Coding[10)] があげられる．Network Coding ではノードの離脱などによりデータを一部損失した場合でも安定して収集できるように，Erasure 符号を利用している．

一方，Structured P2P ではデータの ID に基づいてデータの配置場所が決定される．一般的な Structured P2P の P2P IR では，文書の ID として文書データのハッシュ値を割り当て，文書データは文書の内容と無関係なノードに割り当てられる．この配置方法では，ユーザは問合せに対する適合度の大小にかか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ため，収集する文書数に比例して検索応答に時間がかかってしまう．既存の Unstructured P2P のデータ収集の効率化を目的としたデータ配置法のようにデータを冗長化するにしても，Structured P2P では配置場所が一定のルールに基づいていないとデータを発見しにくいため，この問題を解決できない．これに対して pSearch[20)] では，文書に対して単語ベクトルをもとにした ID を割り当て，検索問合せに適合する文書ほど収集しやすい配置となっている．しかし，pSearch は収集の効率は良いが，単語の転置リストのような索引情報を入手する手間が term-to-document 索引を構築する P2P IR よりも大きくなる．我々が望むような，複数の検索アルゴリズムを P2P IR に適用する検索基盤としては不適切である．文書データの複製を配置するノードを一定のルールで割り当てる手法として，SCOPE[6)] があげられる．しかし，SCOPE は負荷分散には役に立つが，文書データは文書の内容と無関係なノードに割り当てられることには変わりない．

本研究では 1 章で述べたような検索基盤の実現を目指している．分散環境で集中型に比べて検索性能を落とすことなく精度の高い検索を実現するには，term-to-document 索引を構築する P2P IR が適切と考えられる．しかし，先に述べたように，既存研究では文書に対して内容に基づいた ID となっていないため，適合文書の収集効率が低下する．そこで我々は，term-to-document 索引を構築する P2P IR において，検索問合せに適合する文書データを効率良く収集できるようなデータ配置法を検討した．

3. 適合文書収集を効率化するデータ配置手法

3.1 データ配置手法

本研究は，term-to-document 索引を DHT 上で構築する P2P IR を対象とした効率の良い適合文書の収集を目的とする．適合文書を収集するまでの検索応答時間 $T_{s\&g}$ は，次のように計算できる．

$$T_{s\&g} = \underbrace{\sum_{t \in Q} D_{idx}(t)}_{index\ reference} + \underbrace{\sum_{n \in (N_{idx} \cup N_{data})} T_{node}(n)}_{node\ retrieve} + \underbrace{\sum_{d \in RD} D_{data}(d)}_{data\ gathering}, \quad (1)$$

ここで，T_{node} はノードの探索時間，D_{idx} は索引データの転送時間，D_{data} は文書データの転送時間，Q は検索問合せ，t は検索語，RD は適合文書，N_{data} は適合文書を持つノード，N_{idx} は検索語の索引を持つノードを示す．従来手法の問題点は，検索質問に適合する文書が内容に無関係に分散したノードに配置されているため，文書を管理するノードの探索コストが大きいことであった．つまり，式 (1) において $(N_{idx} \cup N_{data})$ の数が検索語数と適合文書数の和とほぼ同値となる問題である．

そこで我々は，文書のデータを検索索引と関連づけて配置する Concordia と呼ぶ手法を提案する．Concordia は，検索時に必ず接続されるノード，つまり，検索質問に含まれる単語の索引を管理するノードにおいて，問合せとの適合度の算出だけでなく適合文書の収集を同時に実行し，文書を管理するノードを探すコストの削減を図る手法である．これを実現できれば，式 (1) において $(N_{idx} \cup N_{data})$ の数を最大で N_{idx} にまで削減できる．このために，Concordia では文書のデータの複製を文書と関連度の高い索引を管理するノードに配置する．文書と索引の関連度の算出には文書における単語の重みを利用する．文書の検索問合せとの適合度は，問合せに含まれる単語の文書における重みの和を用いる[4]．それゆえ，文書における単語の重みに基づいて，文書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，問合せとの適合度の高い文書ほど収集が容易になる．

3.2 システム構成

本章では Concordia の索引構造，索引付け，検索，収集，Erasure 符号のそれぞれについて詳細を述べる．

3.2.1 DHT を用いた索引構造

Concordia では，ノードは DHT 上に索引を作成する．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 DHT で定められた共通のハッシュによって結び付けられており，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．本研究では DHT に Kademlia[13] を採用した．Kademlia を用いることで，ノードは問合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．

Concordia では，この DHT を用いて文書の索引の作成と管理を行う．DHT 上の索引は各単語の索引の集合体である．各単語の索引は，単語のハッシュ値に最も近いノードが管理する．索引には各文書の情報を自由に書き込める．本研究では単語の索引にはその単語を含む文書 1 つにつき，文書名，文書における単語の重み，文書における重みの大きな上位 m 単語のハッシュ値，そして文書のデータへのポインタの合計 4 つの項目を登録した．重みの大きな m 単語のハッシュ値は後に述べるデータ収集の際に利用する．

3.2.2 単語の重みづけと索引への登録

各ノードは索引に文書情報を登録する前に，文書における単語の重みづけの計算を行う．単語の重みづけの計算を終えた後に，各ノードは文書中に含まれる単語の索引に文書名，文書における単語の重み，そして文書における重みの大きな上位 m 単語のハッシュ値を登録する．

この過程で用いる単語の重みづけの式は，Concordia において精度の高い検索と効率の良い収集を実現するうえで，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，単語の重みづけに Okapi で採用された BM25[17] と呼ばれる確率モデルの式を基にした式を用いた[9]．文書 d における単語 t の重みは次のように定義する．

$$w(t,d) = \log \frac{N}{df} \cdot \frac{(k+1) \cdot tf}{k\{(1-\alpha) + \alpha \cdot \frac{dl}{avdl}\} + tf}, \quad (2)$$

ここで，$w(t,d)$ 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．

問合せ Q との適合度は以下の式を用いる．

$$R(Q,d) = \sum_{t \in Q} w(t,d). \quad (3)$$

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では，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 つの値を取得するのが難しい．そこで，我々は 3.2.1 項で述べた索引を用いて，DHT 上に配置された情報を基に 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 を得られる．

文書の総数は動的に変化するため，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．実験より文書の総数が検索精度に与える影響は大きくないことが分かった（4.1 節）．そこで，本研究では簡略化し，適切と思われる定数を設定した．

3.2.3　データ複製と配置

Concordia は，2 種類のデータの複製方法を備えている．1 つは，複製文書をその文章中に現れる重みの大きい単語の索引を保持するノードに配置する方法である．文書を m 個複製して配置するとする．文書中に現れる各単語について式 (2) の値を計算し，そのトップ m 個の単語を $w_1, w_2, \cdots, w_m$ とする．これらの単語の索引を管理するノード $p_1, p_2, \cdots, p_m$ に文書の複製を配置する．以下では，この方法を**単純複製配置法**と呼ぶことにする．**図 1** は，複製数 m を 2 としたときの Concordia における索引と単純複製配置法を示している．図 1 を例に用いて説明すると，“I like birds. I wish for a bird.” という文章 Doc1 の各単語について，まず，DHT 上の索引から得られる df 値を適時用いながら単語の重みの値を計算する．この例では，複製数 m を 2 と設定しているので，先ほどの単語の重みの計算結果のうち，重みの大きな 2 単語，“bird” と “wish” に注目する．それぞれの単語を管理するノードにアクセスし，3.2.2 項で述べたように，文書名や単語の重み，文書において重みの大きい “bird” と “wish” のそれぞれのハッシュ値を，単語の索引に登録する．次に，Doc1 の複製を 2 つ生成し，“bird” と “wish” を管理するノードに転送する．このようにして，単純複製配置法の索引づけとデータ配置を行う．

もう 1 つの方法では，文書の複製の代わりに Erasure 符号を用いて冗長にした n 個の分割データを配置する．Erasure 符号を用いることで，n 個のうち任意の k 個をデコードして文書のデータを得ることができる．Erasure 符号についての詳細は 3.2.5 項で説明する．

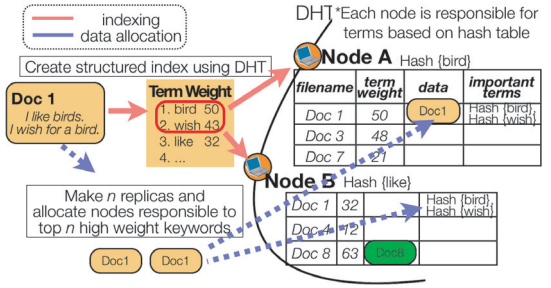

図 1　単純複製配置法の索引づけとデータ配置法
Fig. 1　Indexing and data allocation in simple replica scheme.

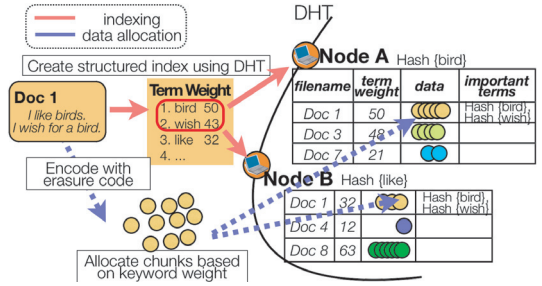

図 2　符号化複製配置法の索引づけとデータ配置法
Fig. 2　Indexing and data allocation in data encode scheme.

Erasure 符号で分割した後，文書中の単語の重みに基づいて分割データを単語の索引を管理するノードに配置する．つまり，文書において重要な単語を管理するノードほど，その文書の分割データをより多く保持することになる．我々はある単語の索引ノードに配置する分割データの数を，文書中の単語の重みの和のうちその単語が占める割合に比例した数に定めた．Erasure 符号を利用することで配置するデータ量を柔軟に調整できるだけでなく，ノードの突如の離脱が頻繁に起こるような環境においても安定して文書のデータを得られる設計になっている．以下では，Erasure 符号を用いたデータの複製配置法を**符号化複製配置法**と呼ぶことにする．**図 2** は，Concordia における索引と Erasure 符号を用いたデータの配置法を示している．

3.2.4　検索とデータの収集

ユーザが問合せに適合する文書を発見・収集するには，まず検索問合せから単語を抽出す

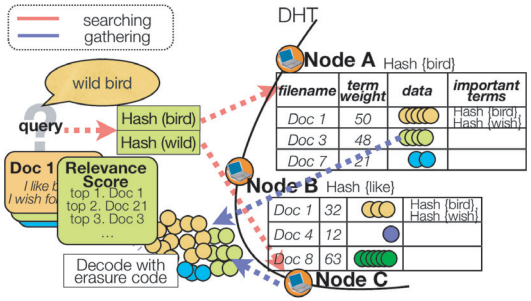

図 3　符号化複製配置法の検索と文書データの収集
Fig. 3　Searching and gathering in data encode scheme.

る．次に，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．そして，索引に登録されている文書それぞれの検索問合せとの適合度を求め，適合度の高い文書を収集する．その際，文書のデータは主に索引を保持するノードから収集する．もし索引を保持するノードに文書のデータが存在しない場合は，索引に登録されたその文書における重要単語 m 個のうちいずれかを参照し，その単語の索引を担当するノードから収集を試みる．符号化複製配置法においては，データ保持ノードから収集した分割データが k 個に満たない場合は，索引に登録されている重みの大きな単語を参照して分割データの収集を続ける．単純複製配置法の検索応答時間は式 (1) となる．一方，符号化複製配置法における検索応答時間 T_{c2} は，式 (1) に分割データのデコード時間が加わり，以下の式で表せる．

$$T_{c2} = T_{s\&g} + \sum_{d \in RD} De(d), \tag{4}$$

ここで，$De(d)$ は文書 d のデコード時間を示す．符号化複製配置における検索と収集法を示したものが **図 3** である．

3.2.5　Erasure 符号を用いた文書の分割法

符号化複製配置法では，文書のデータを Erasure 符号を用いて n 個に分割する．Erasure 符号を用いることで，n 個の分割データのうち任意の k 個から元の文書データを復元できる．つまり，仮に $(n-k)$ 個の分割データ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られる．Erasure 符号には，$(k-1)$ 次関数上の n 個の座標を用いたもの[18] や，k 個の文字列を使った連立方程式を用いたもの[10] 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[21] 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．

$$S = S_0||S_1||S_2||\cdots||S_L$$
$$y = S_0 + S_1 x + \cdots + S_L x^L + r_1 x^{L+1} + \cdots + r_{n-L} x^{k-1} \mod p,$$

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．データの復号に必要な計算量は，ガウスの消去法のアルゴリズムを用いた場合 $O(\log n^3)$，LU 分解を用いた場合 $O(\log n^2)$ 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．

4. 実験結果

Concordia は，既存研究と比べて，P2P 情報検索において問合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．また，Erasure 符号を用いることで，突如のノードの離脱に耐えうるデータの冗長化を目指している．これらの点について評価実験を行った．

実験に使う文書コレクションとして，TREC[3] の Text Research Collection Volume 3 (Tipster 3) を用いた．The San Jose Mercury News (1991), the Associated Press (1990), U.S. Patents (1983–1991), そして Information 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[2] 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．重みづけの式のパラメータは，予備実験の結果をふまえ，k は 100，α は 0.5 を用いた．検索精度は TREC-2 ad hoc & TREC-3 routing topics のトピック 50 件を用いて求めた．適合率と再現率の評価には TREC Eval を使用した[3]．

4.1　総文書数の設定が検索性能へ及ぼす影響

Concordia では文書の検索問合せとの適合度の計算に確率モデルの式 (3) を適用している．式で用いる変数のうち，文書の総数 N と文書の長さの平均値 $avdl$ は処理の簡略化の

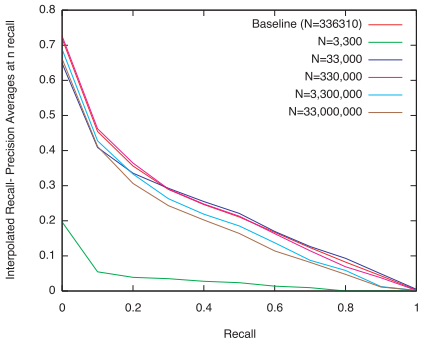

図 4　総文書数の推定値を変化させたときの Interpolated recall-precision average
Fig. 4　Interpolated recall-precision average with regard to the estimated number of docs.

ため定数を設定している．定数を使用することでどの程度検索結果に影響を与えてしまうかについて，実験を行った．実験では，Tipster 3 の文書コレクションに対して TREC-2 ad hoc & TREC-3 routing topics のタスクで検索性能を調べた．

文書の総数 N は式 (2) において，df の値の重みに影響を与える．文書の総数を変化させたときの検索精度を示したものが**図 4** である．図中の Baseline は正確な総文書数を用いて適合度を計算した場合の検索性能である．実験結果から，総文書数 N が適合度の計算に与える影響はさほど大きくないことが分かった．大きく検索性能が低減するときは総文書数が df の値を下回ったときであった．

文書の長さの平均値 $avdl$ が式 (2) に与える影響は，α や k の設定値に依存する．この $avdl$ と α，k は，文書が極端に長いときに文書中の単語の重みを小さくするために用いられる．今回実験に用いた文書コレクションは文書長の差があまりないため，これら変数による影響についての実験は省いた．

以上のことから，単語の重みづけの式において，総文書数や平均文書長など一部の変数を定数に設定することは検索性能にはさほど大きな影響は与えないことが分かった．しかし，精度の高い検索を目指すうえでパラメータチューニングは重要な要素となる．正解データのない実運用環境において重みづけにおけるパラメータの設定は今後の課題と考えている．

4.2　検索実行時に接続するノード数

Concordia における適合文書の収集効率について実験を行った．文書収集の効率は，適合文書の収集過程で拡散する問合せ数で評価した．つまり，式 (1) における $(N_{idx} \cup N_{data})$ の

表 1　配置法ごとの文書の複製数の比較
Table 1　Number of replicas with regard to parameters of redundancy.

比較手法 n	複製文書数	単純複製配置法 n	符号化複製配置法 k/n	複製文書数
1	817,897	1	1.0	336,310
2	1,075,697	2	0.50	672,620
3	1,259,765	3	0.33	1,008,930
4	1,410,761	4	0.25	1,345,240
5	1,542,057	5	0.20	1,681,550
6	1,659,669	6	0.17	2,017,860
7	1,767,310	7	0.14	2,354,170
8	1,867,026	8	0.13	2,690,480
9	1,960,287	9	0.11	3,026,790
10	2,048,181	10	0.10	3,363,100

数で評価する．実験では，TREC-2 ad hoc & TREC-3 routing topics のトピック 50 件それぞれについて，適合度の順位づけを行い，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．ネットワーク内に存在するノード数は簡単のため単語数と同一の値に設定した．本実験では提案手法を，各索引ノードにおいて重みの大きな文書データをキャッシュする手法と比較した．

表 1 は，Tipster 3 を Concordia の 2 つの配置法と比較手法の複製数を比較したものである．単純複製配置法では各文書における重みの大きな上位 n 単語の索引ノードに複製を，符号化複製配置法では各文書における重みの大きな単語の索引ノードに文書の k/n 分割データを，比較手法では各単語の索引中の重みの上位 n 文書の複製を配置した．

単語の索引によっては少ない数の文書しか登録されないものもあるため，比較手法におけるストレージコストの変化は比例関係になっていない．表 1 から，各単語の索引における重みの上位 5 文書を配置するストレージコストの累計は，単純複製配置法において n を 4 から 5 に，もしくは符号化複製配置法において k/n を 0.20 から 0.25 にして分割データを配置するストレージコストの累計とほぼ同じだと分かる．**図 5** は，各手法で複製の配置を行った場合の単語の索引ノードに配置される複製数を比較したものである．図中の Concordia-1 は単純複製配置法を，Concordia-2 は符号化複製配置法を表す．索引ノードにキャッシュする比較手法では索引ノードにほぼ均一の複製数だけ配置されるのに対して，Concordia の配置法の場合では一部の索引ノードへのストレージコストが大きくなってしまうことが分かる．しかし，冗長の度合いを考えると，Concordia の配置法では各文書データが均一な度合いだけ冗長化されるが，比較手法では文書集合に依存するため必ずしも文書間で均一な複製

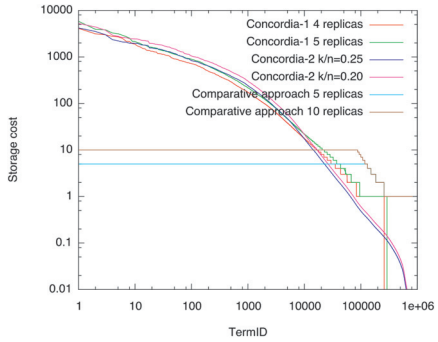

図 5　各単語についてのストレージコストの比較
Fig. 5　Storage cost of each term.

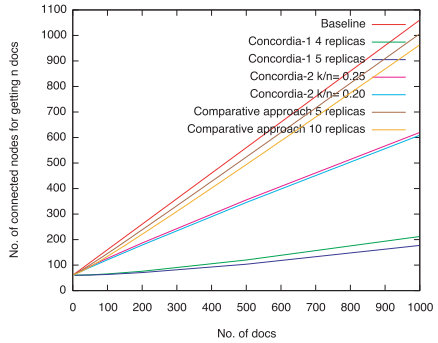

図 6　適合する上位 n 文書の検索・収集時に接続するノード数の比較
Fig. 6　Number of connected nodes for gathering top n relevant docs.

数とならない．そのため，Concordia の配置法では文書データを保全しやすいといえる．

図 6 は，両手法における収集効率を比較したものである．実験結果より，Concordia の配置法の方が文書の収集の過程で接続するノード数は，比較手法よりも少ないことが分かった．さらに，符号化複製配置法よりも単純複製配置法の方が高い収集効率であることが分かった．つまり，冗長化の度合いが等しいとき，文書の収集過程で必要となる問合せ数が少なく，効率の良い収集を実現できることが分かった．

以上のことから，各文書における単語の重みを用いた Concordia の配置法は，ノードに

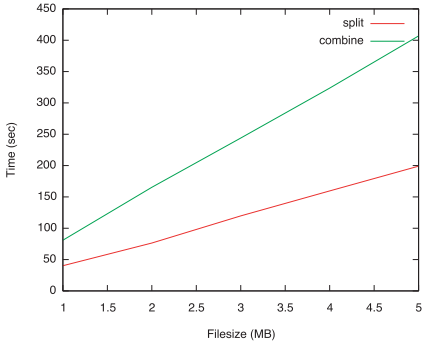

図 7　Erasure 符号の分割と復号の計算時間（(20, 20, 100) 分割）
Fig. 7　Response time for encoding and decomposing data with erasures ((20, 20, 100) partition).

対してのストレージコストを分散する工夫が必要ではあるが，検索問合せとの適合度の高い文書を効率良く収集するには適した配置法であることが確認できた．

4.3　Erasure 符号の有効性

符号化複製配置法では Erasure 符号で十分な数の分割データを生成することで，単語の重みに基づいたデータ配置を実現している．加えて，単純な複製を配置する場合と比べて，ノードの突如の離脱が頻繁に起こるような環境においても安定して文書のデータを得られる設計になっている．本節では Erasure 符号の有効性について考える．

まず，Erasure 符号を用いたデータの分割と復号に必要な時間について実験した．実験に用いたマシンのスペックは，CPU が Core 2 Duo 2.0 GHz で 2 GB のメモリを積んだものであった．Erasure 符号の処理は Python で実装した．復号化のアルゴリズムにはガウスの消去法を用いた．実験結果が**図 7** と**図 8** である．図 7 は分割数を一定にしてファイルサイズを変化させたときの，図 8 はファイルサイズを一定にして分割数を変化させたときの，分割と復号に必要な時間を示している．これらの結果から，ファイルサイズが大きいか，もしくは分割数が多いほど，膨大な分割と復号に時間がかかることが分かった．つまり，大きなファイルを扱う P2P IR には Erasure 符号による複製は不向きであるといえる．また，Erasure 符号の分割数は P2P IR に参加するノードのマシンスペックを考慮する必要があると分かった．Erasure 符号は復号の計算コストが高いため，効率的な検索処理を実現するためには，現在の PC のさらなる計算処理能力の向上が望まれる．

次に，Erasure 符号を用いた冗長化の性能についての実験を行った．ノードが頻繁に離脱す

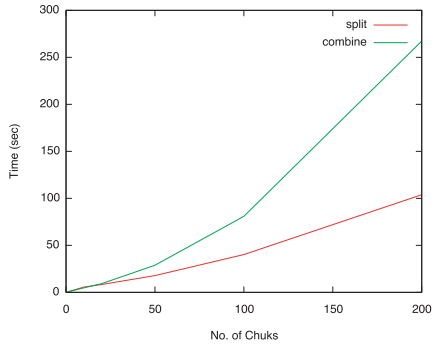

図 8　Erasure 符号の分割と復号の計算時間（1 MB のファイルを (N/5, N/5, N) 分割）
Fig. 8　Response time for encoding and decomposing data with erasures ((N/5, N/5, N) partition).

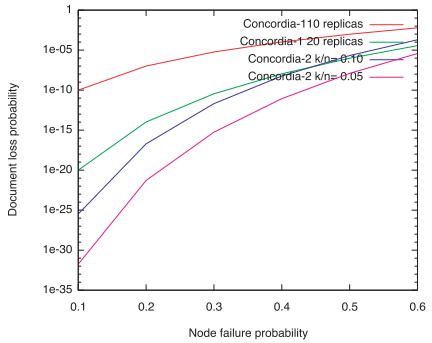

図 9　文書のデータの損失確率（ZF32-345-1050）
Fig. 9　Document Loss Probability (ZF32-345-1050).

る状況で文書の収集がどの程度困難になるか，単純な複製を用いた場合の冗長化と Erasure 符号を用いた冗長化とでシミュレーションを用いて比較した．Erasure 符号における文書のデータの損失確率は分散情報の配分，つまり文書中の単語の重みに依存する．実験には，Tipster 3 に含まれる文書番号 ZF32-345-1050 を使用した．単純な複製を配置する手法と符号化複製配置法において，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 9** である．図中の Concordia-1 は単純複製配置法，Concordia-2 は符号化複製配置法を表す．冗長化の度合いが同じとき，Erasure 符号を用いたデータ複製のほうが文書全体を複製する手法と比べて文書を安定して取得できることが分かった．つまり，符号化複製配置法は Erasure 符号を用いることにより，単純な複製を用いる手法よりも効率の良い冗長化が実現できていることが分かる．

4.4　Erasure 符号を用いたデータ配置法の検索応答時間

符号化複製配置法を用いた場合の検索応答時間は式 (4) で示せる．4.2 節に示した比較手法における検索応答時間を T_{xxx} とおくと，符号化複製配置法による検索応答時間の削減 T_{reduce} は，以下の式で概算できる．

$$\begin{aligned} T_{reduce} &= T_{c2} - T_{xxx} \\ &= (N_{c2} - N_{xxx}) \times T_{node} - \sum_{d \in RD} De(d), \end{aligned} \tag{5}$$

ここで，各複製配置法で検索実行時に接続するノード数，つまり式 (1) における $(N_{idx} \cup N_{data})$ の数をそれぞれ N_{c2} と N_{xxx} とおいた．仮に，検索問合せと適合度の高い 300 文書を収集する場合，図 6 から $(N_{c2} - N_{xxx})$ はおよそ 100 ノードの探索時間を削減できると分かる．一方，4.3 節によると，$k/n = 0.20$ の 1 MB の分割データをデコードするのに 40 秒ほど必要だと分かる．これより，300 文書すべてをデコードするのに必要な時間は，現在の計算能力では 100 ノードの探索時間よりも明らかに大きくなってしまうと思われる．

Concordia のデータ配置法によって検索時に探索するノード数は大幅に削減できるが，符号化複製配置法では Erasure 符号の復号化コストの改善が必要である．特に冗長化の必要がなく収集処理効率を重視する場合は，単純複製配置法が適しているといえる．

5. 結　　論

本稿では，P2P IR における文書の新しい分散配置手法である Concordia を提案した．Concordia は，文書を単語の重みに基づいて配置し，問合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 は，データの複製法として，処理性能を重視した単純複製と Erasure 符号を用いた符号化複製モードを有している．符号化複製配置法は，ノードの突如の離脱が起こりうる状況でも安定して文書を収集できる P2P IR システムである．実験から，Concordia の配置法にすることで問合せに適合する文書ほど索引参照時に接続するノードから収集でき，文書を管理するノードを探すコストを削減できることが分かった．つまり，P2P 情報検索システムのデータ配置法に単語の頻度情報を利用することで，適合文書収集の効率化を実現できた．

現在は，Concordia を文書の収集をともなう検索支援システム[24),25)] とマッシュアップさ

せた P2P IR のアプリケーションを実装している．また，P2P 環境で検索基盤を構築することを目標に，索引構築の負荷分散手法に取り組んでいる．この負荷分散手法を Concordia に応用して，提案手法における課題であるストレージコストの分散を実現したいと考えている．

参 考 文 献

1) Gnutella. http://www.gnutella.com/
2) Porter stemmer. http://www.tartarus.org/martin/PorterStemmer/
3) Text REtrieval Conference (TREC). http://trec.nist.gov/
4) Baeza-Yates, R. and Ribeiro-Neto, B.: *Modern Information Retrieval*, ACM Press (1999).
5) Bender, M., Michel, S., Triantafillou, P., Weikum, G. and Zimmer, C.: MINERVA: Collaborative P2P Search, *Proc. VLDB'05* (2005).
6) Chen, X., Ren, S., Wang, H. and Zhang, X.: SCOPE: Scalable Consistency Maintenance in Structured P2P Systems, *Proc. INFOCOM'05* (2005).
7) 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).
8) 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, *Proc. HPDC'03* (2003).
9) Fang, H., Tao, T. and Zhai, C.: A Formal Study of Information Retrieval Heuristics, *Proc. SIGIR'04*, pp.49–56 (2004).
10) Gkantsidis, C. and Rodriguez, P.: Network Coding for Large Scale Content Distribution, *Proc. INFOCOM'05* (2005).
11) Huang, J., Li, X. and Wu, J.: A Class-Based Search System in Unstructured P2P Networks, *Proc. AINA'07* (2007).
12) Kurasawa, H., Wakaki, H., Takasu, A. and Adachi, J.: Data Allocation Scheme Based on Term Weight for P2P Information Retrieval, *Proc. WIDM'07* (2007).
13) Maymounkov, P. and Mazieres, D.: Kademlia: A Peer-to-Peer Information System Based on the XOR Metric, *Proc. IPTPS'02* (2002).
14) Michel, S., Bender, M. and Ntarmos, N.: Discovering and Exploiting Keyword and Attribute-Value Co-occurrences to Improve P2P Routing Indices, *Proc. CIKM'06* (2006).
15) Mueller, W., Eisenhardt, M. and Henrich, A.: Scalable Summary Based Retrieval in P2P Networks, *Proc. CIKM'05* (2005).
16) Podnar, I., Rajman, M., Luu, T., Klemm, F. and Aberer, K.: Scalable Peer-to-Peer Web Retrieval with Highly Discriminative Keys, *Proc. ICDE'07* (2007).
17) Robertson, S.E., Walker, S., Jones, S., Beaulieu, M.M.H. and Gatford, M.: Okapi at TREC-3, *Proc. TREC-3*, pp.109–126 (1994).
18) Shamir, A.: How to Share a Secret, *Comm. ACM*, Vol.22, No.11 (1979).
19) Suel, T., Mathur, C., Wu, J., Zhang, J., Delis, A., Kharrazi, M., Long, X. and Shanmugasundaram, K.: ODISSEA: A Peer-to-Peer Architecture for Scalable Web Search and Information Retrieval, *Proc. WebDB'03* (2003).
20) Tang, C., Xu, Z. and Dwarkadas, S.: Peer-to-Peer Information Retrieval Using Self-Organizing Semantic Overlay Networks, *Proc. SIGCOMM'03* (2003).
21) Yamamoto, H.: On Secret Sharing Systems Using (k,L,n) Threshold Scheme, *IECE Trans.*, Vol.J68-A, No.9, pp.945–952 (1985).
22) Yang, K. and Ho, J.: Proof: A DHT-Based Peer-to-Peer Search Engine, *Proc. WI'06* (2006).
23) Zhu, Y., Yang, X. and Hu, Y.: Making Search Efficient on Gnutella-Like P2P Systems, *Proc. IPDPS'05* (2005).
24) 若木裕美，正田備也，高須淳宏，安達　淳：検索語の曖昧性解消のためのトピック指向単語抽出および単語クラスタリング，情報処理学会論文誌：データベース，Vol.47, No.SIG 19 (TOD 32), pp.72–85 (2006).
25) 辻下卓見，相澤彰子，高須淳宏，安達　淳：情報爆発時代のための制約付クラスタリングを用いた適合性フィードバック手法の提案，情報処理学会全国大会 (2008).

(平成 20 年 3 月 19 日受付)
(平成 20 年 6 月 10 日採録)

(担当編集委員　井上 潮)

倉沢　央（学生会員）

2006 年東京大学工学部電子情報工学科卒業．2008 年同大学大学院情報理工学系研究科電子情報学専攻修士課程修了．同年から東京大学大学院情報理工学系研究科電子情報学専攻博士課程に在籍．分散情報検索システム，コンテキストアウェアネットワークの研究に従事．電子情報通信学会，日本データベース学会各会員．

若木　裕美（正会員）

2002 年東京大学工学部電子情報工学科卒業．2004 年同大学大学院新領域創成科学研究科基盤情報学専攻修士課程修了．2007 年同大学院情報理工学系研究科電子情報学専攻博士課程修了．博士（情報理工学）．同年（株）東芝入社．現在東芝研究開発センターにてテキストマイニングの研究に従事．

高須　淳宏（正会員）

1984 年東京大学工学部航空学科卒業．1989 年同大学大学院工学系研究科博士課程修了．工学博士．同年学術情報センター研究開発部助手．同センター助教授．国立情報学研究所助教授を経て 2003 年より同研究所教授．データ工学，特にデータ解析と解析モデルの学習の研究に従事．電子情報通信学会，人工知能学会，日本データベース学会，ACM，IEEE 各会員．

安達　　淳（正会員）

1981 年東京大学大学院工学系研究科博士課程修了．工学博士．東京大学大型計算機センター助手，文部省学術情報センター助教授，教授等を経て，現在，国立情報学研究所教授．東京大学大学院情報理工学研究科教授を併任．データベースシステム，テキストマイニング，情報検索，電子図書館システム等の研究開発に従事．電子情報通信学会，日本データベース学会，IEEE，ACM 各会員．