社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2008－MPS－72 (40)
2008－BIO－15 (40)
2008／12／18

蟻メタファーを用いたブロガーの興味推移の視覚化

湊 匡平[†1] 林 貴宏[†1] 尾内 理紀夫[†1,†2]

本稿では，ブロガーと蟻を対応付け，ブロガーの興味の推移を蟻が餌から餌へと移動していくというメタファーで表現し，ユーザの新たな興味発見を促すシステム「ARIBROS」について述べる．「ARIBROS」では，ユーザが任意のブログ記事のURLを入力することによって，そのユーザの興味を推定し，興味の似ているブロガーを蟻として表示する．また，表示されたブロガーが興味のあるキーワードを蟻の餌として表示する．蟻は対応しているブロガーの興味の推移に応じて，餌から餌へと移動していく．この様子を観察することによって，ユーザは長期的な興味の一致を持つブロガーを発見することができる．

Visualizing Transition of Bloggers' Interests by Using a Metaphor of Ants

Kyouhei Minato,[†1] Takahiro Hayashi[†1] and Rikio Onai[†1,†2]

This paper reports on ARIBROS: a system for recommending bloggers to a user. When the user posts a blog to the system, the system estimates the user interests and recommends bloggers who have similar interests to the user. The user can discover his /her new interests by the recommendation. The system has a feature that transition of bloggers' interests is represented with the metaphor of motion of ants. Bloggers are represented as ants. Keywords which express bloggers' interests are represented as foods. Ants move between foods according to transition of bloggers' interests. The user can discover new interests by observing motion of ants.

1. はじめに

近年のブログの書き手(以下，ブロガー)の爆発的な増加に伴い，ブログ全体が持つ情報量が膨大になってきている．それらの情報は有用であるが，情報量が膨大すぎるために，必要な情報を発見するのは非常に困難な状態である．必要な情報を効率よく得る方法として，キーワードなどを用いたブログ検索システムを利用する方法が存在するが，新たな興味が未知の段階においてキーワードを決定することが困難なため，新たな興味となるようなブログを発見することは難しい．

このような未知のものを検索する手段として，システムがユーザの嗜好を読み取り，システム側から情報を推薦するという情報推薦というものが存在する．これらは，協調フィルタリング[1]やコンテンツベースフィルタリング[2]といった手法により実現されている．協調フィルタリングは，ユーザの閲覧履歴情報などを利用して，ユーザの嗜好と類似した他のユーザの情報を用いて推薦する情報の選択を行う手法である．コンテンツベースフィルタリングでは，ユーザの閲覧履歴情報などから取得したユーザの特徴表現とコンテンツごとの特徴表現を比較して，推薦するコンテンツを決定する手法である．

これまでに，ユーザにブログ記事を推薦するシステムは森本ら[3]や岸田ら[4]などによって開発されている．しかし，ブログ記事はブロガーの書いたある時点での興味であり，一つの記事だけではブロガーの短期的な興味しか表せていないことが多い．つまり，ブログ記事を推薦しただけでは，ユーザとブロガーの短期的な興味の一致しか考慮できないのである．そこで，本稿ではより長期的な興味の一致をユーザに発見させ，より強い興味喚起をユーザに起こさせる方法を提案する．本システムでは，ブログ記事ではなくブロガーを推薦する．そして，ユーザが推薦されたブロガーの興味の推移を観察することによって，長期的な興味の一致を発見するようにユーザに促す．また，ユーザと興味の似ているブロガーを推薦し，そのブロガーの他の興味対象もユーザは見ることができるため，協調フィルタリングの効果も期待できる．

さらに，そのブロガーの興味の推移を蟻が餌から餌へと移動していくというメタファーを利用して表現す

†1 電気通信大学電気通信学部
Department of Computer Science, The University of Electro-Communications
†2 電気通信大学大学院電気通信学研究科
Graduate School of Electro-Communications, The University of Electro-Communications

る．ここで，ブロガーを蟻に対応させ，ブロガーの興味対象を蟻の餌に対応させる．このことにより，ユーザはより直観的にブロガーの興味の推移を観察することができる．

以降，第2章でシステムの概要について，第3章で推薦部の処理について，第4章でデータベースについて，第5章でインターフェース部について，第6章で考察を行い，第7章でまとめる．

2. システムの概要

2.1 システム構成

本システムの構成図を図1に示す．本システムは，大きく分けて推薦部とデータベース，インターフェース部から構成される．

推薦部はユーザが入力したブログ記事URLを解析し，ユーザの興味を推定し，そして，データベースに保持されている過去のブログ記事データから推薦する情報を決定する処理を行う．また，Webサーバーには Tomcat [*1]を用いている．これは，Javaで書かれている Tomcat を利用することによって，形態素解析器 Sen [*2]を利用でき，JDBC を用いればデータベースを操作することも可能となるためである．

データベースは過去のブログ記事データを格納する．本システムではデータベースを管理するシステムとして MySQL [*3]を用いている．MySQL を選択した理由は，低コストであること，世界的に Web サイトの構築に用いられているという実績があること，資料が豊富なことが挙げられる．

インターフェース部は，Adobe Flash [*4]を用いて作成されている．これは，蟻が餌から餌へと移動していくという動きをアニメーションで表示するため，アニメーション表示によく用いられる Flash を用いて作成するのが有効であると考えたためである．また，Flash は HTTP 通信を利用して，Web サーバーとテキストデータや XML データの通信を行うことができる．本システムでは，これを利用し Flash はユーザから入力されたブログ記事 URL を Web サーバーに送信し，サーバー側で処理した結果の XML データを受信する．

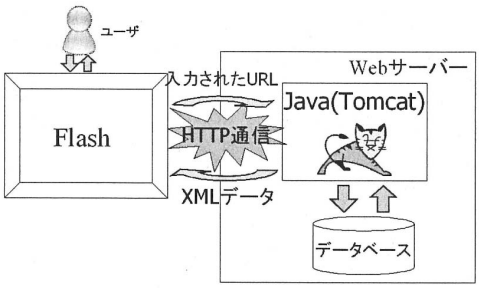

図1 システム構成図
Fig. 1 System Configuration

3. 推薦部の処理

3.1 推薦部の処理の流れ

推薦部ではユーザが入力したブログ記事の URL から推薦する情報を決定する．

推薦部の処理の流れは次のようになっている．

(1) 入力された URL の HTML を取得し，その HTML を解析する．
(2) 解析された HTML のブログ本文部分を推定し，抽出する．
(3) ブログ本文を形態素解析し，索引語を抽出する．
(4) 索引語を元にデータベースから関連するブログ記事を取得する．
(5) それぞれの記事に対してスコア付けする．
(6) スコア上位の記事のブロガーと，それと関連するキーワードをデータベースから取得する．
(7) 取得したデータを XML にして，Flash に XML データを送信する．

3.2，3.3 節では，(2) と (5) の部分に関して，本文抽出方法と記事へのスコア付け方法について詳しく説明する．

3.2 本文抽出

取得した HTML からブログ本文になる可能性のある部分を取り出す．ここでは，本文になる可能性の高い div，td タグで囲まれた範囲をテキストブロックとして取り出す．このとき，50 文字以下の文はリンクやコメントなどの可能性が高いため，除外することとしている．次に，テキストブロックごとにテキスト長と句読点の数から本文らしさを表すスコアを算出する．また，ブログ本文の多くは Web ページの最初の方に出現するため，後半のブロックほど，スコアが低くなるように減衰係数を設定する．n 番目のテキストブロックのテキスト長を l_n，句読点数を p_n とすると n 番目のテキストブロックのスコア S_n は，

$$S_n = (l_n + p_n \times W) \times D^n \quad (1)$$

⋆1 Apache Tomcat, http://tomcat.apache.org/
⋆2 Sen, https://sen.dev.java.net/
⋆3 MySQL, http://www.mysql.com/
⋆4 Adobe Flash,
http://www.adobe.com/jp/products/flash/

で定義される．ここで，W は句読点につける重みで，本システムでは $W = 10$ に設定している．また，D は減衰係数で，本システムでは $D = 0.8$ としている．

その後，スコアが閾値以上で連続するブロックは，一つのブロックにまとめる．ブロック中でスコア最大のブロックを本文として抽出する．ここで，閾値は 100 とした．ブロックを一つにまとめる理由は，本文に該当する部分が div や td タグで写真の挿入や段組みなどのために区切られていたとしても，それらが連続していることを反映するためである．

3.3 記事へのスコア付け

記事へスコア付けする前に抽出した索引語を検索クエリとしてデータベースへアクセスする必要があるが，クエリの数が多いとそれだけデータベース内の探索に時間がかかってしまう．そこで，ブログ本文の特徴的な部分のみを取り出す目的で，索引語は代名詞以外の名詞と未知語の 2 種類のみとする．さらに，検索クエリとするのは，索引語のうちブログ本文中で頻度の高いものから 10 語とする．

データベースから取り出したブログ記事に対するスコアは TF・IDF を基本として計算される．ブログ記事 D_j と索引語 w_i との間のスコア s_{ij} は，

$$s_{ij} = f_{ij} \times \log \frac{N}{DF_i} \tag{2}$$

で定義される．ここで，f_{ij} は D_j に索引語 w_i が含まれれば 1，そうでなければ 0 となる変数である．N はブログ記事集合全体のブログ記事数，DF_i は索引語 w_i が含まれるブログ記事数である．ブログ記事 D_j のスコア s_j は，

$$s_j = \sum_{i=1}^{N} (s_{ij} \times F_i) \tag{3}$$

で定義される．ここで，F_i はユーザが入力したブログ記事に含まれる索引語 w_i の頻度である．このようにして，ブログ記事のスコアを求め，スコアの高いブログ記事のブロガーを推薦対象とする．

4. データベース部

4.1 データベースに格納するデータ

データベースに格納されているデータは，ブログ記事の URL，タイトル，日付，カテゴリ，著者，その記事内のキーワードとそのタイプ (最大 10 組)，ブログサイトの URL，ブログサイトのタイトルである．これらのデータは，2007 年 1 月 1 日〜12 月 31 日にソネットエンタテインメント株式会社 (以下，So-net) が収集したもので，リコメンデーションサービスコンテスト[*1] に参加することで，提供されたものである．本システムでは，これらのデータは RDBMS として MySQL を利用して管理し，データベース化している．

4.2 テーブルの設計

4.1 で述べたデータを本システムでは，表 1 のように article，keyword，blog の 3 種類のテーブルを用意して保存する．ここで，PK は主キーであることを表す．

4.3 データベースへのアクセス

データベースへは Web サーバーである Tomcat からアクセスされるが，このとき Prepared Statement(準備済み SQL 文) を使用してデータベースへのアクセスを行う．これにより，データアクセスの効率化を図ることができる．

5. インターフェース部

5.1 蟻メタファー

1 章で述べたように，ブロガーの興味の推移に応じて興味が変化していく様を，蟻が餌から餌へと動いていくというメタファーを利用して表現する．ここで，蟻はブロガー，餌がキーワードを表す．このようにすることで，ブロガーごとの興味の推移を見ることができる．また，このインターフェースの画面イメージを図 2 に示す．この画面では，蟻が餌から餌へと移動していく．蟻の上に表示されている文字列は，その蟻に対応させたブロガーの情報である．表示する情報はブロガー名とそのブロガーのブログサイトのタイトルである．また，餌には対応させたキーワードの情報を表示し，その内容はキーワードとそのタイプである．タイプとは，キーワードの意味属性で場所なら LOCATION，組織なら ORGANIZATION などである．

5.2 ユーザ操作

次に，ユーザが可能な操作について述べる．各種の操作は可能な限り簡単にし，ユーザの情報取得コストを抑えるようにしている．実際に実装している具体的な機能は次の 5 つである．

- ユーザ自身が動かす蟻を導入する．他の蟻と接触することにより，餌を分けてもらえる (接触した蟻に関連するキーワードを見ることができる)．また，餌を貰うと，それに対応するキーワードに興味を持つ蟻 (ブロガー) が寄ってくる．
- 蟻をクリック&ドロップし，URL 入力のテキストフィールドに持っていくと，その蟻 (ブロガー) の

*1 リコメンデーションサービスコンテスト, http://www.ipsj.or.jp/sig/mps/compe08html_release/contest.html

article[ブログ記事テーブル]	
PK	id[記事 ID]
Index	time[取得日時]
	URL[記事 URL] title[記事のタイトル] category[記事のカテゴリ] author[記事の著者]

keyword[キーワードテーブル]	
PK	id[キーワード ID]
Index	word[キーワード]
	type[キーワードのタイプ]

blog[ブログサイトテーブル]	
PK	id[ブログサイト ID]
Index	blogURL[ブログサイトの URL]
	blogTitle[ブログサイトのタイトル]

表 1 データベースのテーブル

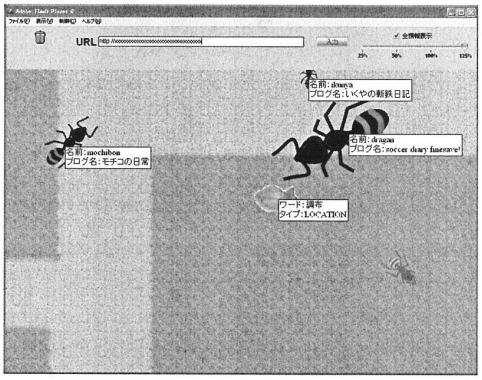

図 2 画面イメージ
Fig. 2 System Image

ブログ記事の URL が入力され，新たな入力とすることができる．

- 興味のある蟻がいた場合，それをクリックすることによりチェックすることができる．チェックされた蟻は，色が変化し，一目で分かるようになる．
- 興味が無いと判断した蟻は取り除くことができる．ただし，完全に除外するのではなく，あとで元に戻すこともできる．
- 先に述べた蟻のチェック，削除によりチェック，削除された蟻との関連性に応じて，他の蟻も色がだんだんと変化していく．

これらの操作により，ユーザは自身の興味発見の手助けをすることができる．例えば，興味のあるブロガーにチェックをつけておくことによって，ユーザは興味のあるブロガーを見失うことなく興味の推移を観察できる．これは，長期的な興味の一致を発見させることが目的である本システムでは，非常に重要である．

6. 考　　察

本システムでは，ブログ記事ではなくブロガーを推薦する．ブロガーを推薦し，その興味の推移を見せることによって，長期的な興味の一致をユーザに発見させることができると期待される．これにより，ユーザにブログ記事推薦のような短期的な興味の一致よりも強い興味を喚起することができると期待される．さらに，ユーザが興味を持つ新たなキーワードを発見したときに，それに興味を持つブロガーを提示できるようにすることで，提示されたブロガーの興味の推移を観察でき，ユーザのさらなる興味発見へとつながると期待できる．

また，人気のあるブロガーは，人に興味を起こさせる文章を書くことが多い．そこで，ブロガーの人気度といった指標を導入すれば，より有効な推薦が可能になると考えられる．ブログサイトのアクセス数やブログ記事のコメント数，トラックバック数などの情報を利用することにより，ブロガーの人気度を評価することができれば，人気のあるブロガーを優先的に推薦することも可能になり，より質の高い推薦が期待できる．これは，今後の課題である．

7. おわりに

本稿では，ブロガーと蟻を対応付け，ブロガーの興味の推移を蟻が餌から餌へと移動していくというメタファーで表現し，ユーザの新たな興味発見を促すシステム「ARIBROS」について述べた．今後の課題として，本システムに関する被験者実験を行い，ブロガーの興味推移を視覚化することによる有効性を検証する．さらに，ユーザに興味を起こさせるブロガーをより多く発見するために，ブロガーの人気度評価手法をシステムへ導入し，その有効性を検証したい．

参 考 文 献

1) 清水拓也，土方嘉徳，西田正吾：発見性を考慮した協調フィルタリングアルゴリズム，電子情報通信学会論文誌 D， Vol.J91-D, No.3, pp.538–550 (2008).
2) 久津見洋，内藤榮一，荒木昭一，江村里志，新居薫治：ユーザ適応型ホームページ推薦ソフトウェブナビゲーターの開発，電子情報通信学会論文誌 D-II，Vol.J84-D-II, No.6, pp.1149–1157 (2001).
3) 森本和伸，林　貴宏，尾内理紀夫：MineBlog:興味発見を支援する blog 記事推薦システム，情報処理学会論文誌， Vol.47, No.4, pp.1171–1180 (2006).
4) 岸田真和，倉本　到，渋谷　雄，辻野嘉宏：ウェブログにおけるユーザの興味のあるエントリの推薦法，電子情報通信学会技術研究報告. HIP, ヒューマン情報処理，Vol.104, No.746, pp.25–30 (2004).