特別研究報告

題目

協調フィルタリングを用いたソフトウェア部品推薦手法の提案と実装

指導教官
井上 克郎 教授

報告者
市井 誠

平成 16 年 2 月 19 日

大阪大学 基礎工学部 情報科学科

平成 15 年度 特別研究報告

協調フィルタリングを用いたソフトウェア部品推薦手法の提案と実装

市井 誠

内容梗概

近年のソフトウェアの大規模化と複雑化に伴い，ソフトウェア部品の再利用の為の技術が必要とされており，その一つとしてソフトウェア検索システムがある．我々の研究チームでは，Java のソースコードを対象としたソフトウェア部品検索システム SPARS-J (*Software Product Archive, analysis and Retrieval System for Java*) の構築を行なっている．SPARS-J では，キーワードによる部品検索の他に，部品同士の利用関係やパッケージ階層を辿る事により部品を取得する事ができる．これらはソースコード，つまり部品そのものを解析して得られる情報を基にしている．一方，利用者の検索履歴を利用する事で，同時に使用される部品といった部品間の情報を得る事ができる．これらの情報は，部品そのものを解析して得られる物ではないため，検索履歴を利用する事で，より有効な部品の検索を行う事ができると考えられる．

そこで本研究では，ソフトウェア部品検索システムに対する協調フィルタリング手法を提案し，SPARS-J に実装した．協調フィルタリング手法は，オンラインショップなどで個々の利用者に合わせた推薦を行うための手法である．ソフトウェア部品検索に適用する事で利用者の目的に応じた部品の推薦を行う事ができると考えられる．実現したシステムでは，利用者の検索履歴をデータベースに記録し，蓄積された情報に対して協調フィルタリング手法を適用する事で個々の利用者に必要と思われる部品を推薦する．そして実現したシステムをもとに適用実験を行い，推薦機能を利用する事で検索効率が向上する事を確認した．

主な用語

ソフトウェア部品 (Software Component)
ソフトウェア検索 (Software Retrieve)
協調フィルタリング (Collaborative Filtering)

目 次

1　まえがき

ソフトウェアの大規模化と複雑化に伴い，高品質なソフトウェアを一定期間内に効率良く開発することが重要になってきている．これを実現するために，近年のソフトウェア開発において再利用を用いた開発がよく行われている．再利用とは，既存のソフトウェア部品を同一システム内や他のシステムで利用することを指し，開発期間の短縮や品質向上を期待できるといわれている [14, 2, 10, 1]．ソフトウェアの再利用による効果を最大限に引き出すためには，開発者が開発しようとするソフトウェアに必要な部品およびライブラリに関する知識を持つことが重要になってくるが，知識の共有が満足になされていないために，同種のプログラムが別々の場所で，独立して開発されている事も多い．

一方でインターネットの普及により，SourceForge[11] などのソフトウェアに関する情報を交換するコミュニティが誕生し，大量のプログラムソースコードが簡単に入手できるようになった．これらの公開されている大量の部品の中から，開発者の必要としている機能を持つ部品，その機能の使い方を示している部品のような，再利用に有益な情報を提供する検索システムを実現する事で，知識の共有が実現でき，再利用を促進する事ができると考えられる．

我々の研究チームではソフトウェア部品の収集，検索システム *SPARS* (Software Product Archiving, analyzing, and Retrieving System) を研究しており， Java プログラムを対象として SPARS-J という検索システムを開発している．SPARS-J は，依存や類似といったソフトウェア部品特有の特性を考慮しながら大規模なライブラリの分類・検索を自動的に行なうシステムである．キーワードによる全文検索を行ない，ソフトウェアのソースコードを効率良く検索することが可能である．さらに，検索結果表示の際にソフトウェア部品に関する詳細な情報を併せて提供する．このシステムを用いることで，ライブラリの知識が無い開発者も有用なソフトウェア部品やそれに付随する有益な情報を容易に入手することができる．

SPARS-J では，ソースコード中の文字列を切り出した索引語や，クラス同士の関連を解析した利用関係，またパッケージ階層など，部品そのものから得られる情報のみを基にして検索を行う．一方，利用者の検索行動を追う事により，頻繁に使用される部品や同時に使用される部品といった情報が取得可能であると考えられる．これらの情報を SPARS-J の部品検索に利用する事で，検索者はより有用な結果を得る事ができると考えられる．

そこで，本研究では利用者の検索履歴を利用する手法として協調フィルタリングによるソフトウェア部品の推薦を行う手法を提案し，SPARS-J 上で実現する．協調フィルタリングとは，Netnews の記事や映画，音楽などの推薦システム [8] において用いられる手法であり，「自分と好みがにている人が好む物は，自分も好むだろう」という直観的な仮定に基づいている [9]．この手法では，まずユーザからアイテムに対する評価を取得して蓄積し，その評

価の傾向からユーザの好みを推測する．そして推薦対象のユーザと好みが類似しているユーザが高く評価したアイテムを推薦する．ソフトウェア部品検索システムにおいては,「好み」を「検索目的」と置き換えて適用する事で，利用者の目的に応じた部品の推薦を行う事ができると考えられる．また，実現したシステムの適用実験を行い，その有効性を検証する．

以下，2 節で SPARS-J について説明する．3 節では，協調フィルタリング技術に関する関連研究を述べた上で既存の代表的な手法を説明する．4 節で協調フィルタリング手法のソフトウェア部品の推薦のための拡張を提案し，提案手法の SPARS-J への実装について述べる．5 節で適用実験を行い，その有効性を検証する．最後に 6 節でまとめと今後の課題について述べる．

2　Java ソフトウェア部品検索システム SPARS-J

2.1　準備

本節では，ソフトウェア部品や類似部品群，検索結果の順位付け手法など，SPARS-J の根幹をなす概念について簡単に説明する．

2.1.1　ソフトウェア部品と部品群

一般にソフトウェア部品 (Software Component) は再利用できるように設計された部品とされる [5]．しかし，部品の集合にはコピーした部品や，コピーして一部変更した部品が多く存在する．そこで，SPARS-J では類似した部品をまとめることにより，部品の集合をいくつかの部品群に分類する．部品の集合を部品群に分類するために，任意の部品間の類似度 (Similarity) をメトリクスを用いて定量的に評価し，閾値以上の類似度をもつ部品を同一部品群として分類する．また一般的に，単体の部品間には互いに利用する，利用されるという利用関係が存在する．そこで，ある部品群に属する部品が，他の部品群に属する部品を利用している場合には，その 2 つの部品群間には利用関係が存在するとみなしている．

2.1.2　順位付け手法

一般に部品検索をキーワードによって行なうと，検索キーと合致する索引キーを持つものが多数存在する場合がある．そのため，検索結果を提示する際に適当な順位で表示することが必要となる．テキスト文書を対象とした情報検索システムで一般的に用いられている手法では，検索対象となる文書集合から各文書の特徴を表すキーワードである索引語を抽出し，索引語の集合によってその文書の内容を近似する．登録するそれぞれの文書の特徴を的確に表すように付与された，索引語の集合などからなる情報のことを索引キーと呼ぶ．検索キーは検索者の要求の内容を近似しているため，検索キーと索引キーを用いて検索キーと文書の適合度を測り，順位付けするのが一般的である．しかし，ソフトウェア部品を対象として再利用のための部品を検索する場合は，検索者の要求の内容と適合する部品であるかどうかのほかに，その部品がよく利用されている部品かどうかに関しても考慮する必要がある．検索キーと適合度が高い部品よりも，よく利用されている部品を上位に提示した方が，利用例も多く出る可能性があり，部品の再利用をスムーズに行なうことが可能な場合もある．そのため，利用しやすい部品かどうか定量的に評価するための指標を導入し，検索キーと部品の適合度も併せて，両面を考慮した部品の順位付けを行なう必要がある．

索引語の重み付けによる適合度の評価　索引語の中には部品の内容と密接に関係したものもあれば，関係の薄いものも存在する．抽出された索引語が部品の内容を表すうえでどれだ

けの重要度を持っているか測ることができれば，より精度の高い検索を実現できると考えられる．このために用いられるのが索引語の重み付けである．索引語の重みを利用することによって，同じ索引語を含む部品でもその索引語の各部品中での重要度を考慮して，検索キーに対する部品の適合度を計算し部品を順序付けすることが可能になる．検索者が与えた検索キーがある部品の索引キーにヒットしたとき，その索引キー中の索引語の重みの総和を検索キーと部品の適合度とする．SPARS-J における索引語の重み付け手法として，情報検索の分野で一般的に用いられる TF-IDF 法 [15] を用いて各部品における重み付けを行なう．TF-IDF 法は，任意の部品中における特定の索引語の出現頻度 TF(Term Frequency)，および特定の索引語を含む部品数の逆数 IDF(Inverse Document Frequency) の値を正規化して重みを算出する．TF は部品内で出現頻度の低い索引語と高い索引語を差別化し，部品をより特徴付ける語を選別するためのものである．IDF は部品集合内の他の部品の索引語の分布について考慮するためのもので，ある索引語が，どの程度その部品に特徴的に現れるのかという特定性を示す．検索キーと部品の適合度の高い順に順位付けすることを，後述する CR 法に対して KR 法 (Keyword Rank 法) と呼ぶことにする．また，測定した適合度の値を KR 値と呼ぶ．

利用関係による評価　我々の研究チームではこれまでに，利用関係からソフトウェア部品の利用実績を測定し，順位付けし，評価する手法 (Component Rank 法，CR 法) を提案している [17]．CR 法では，十分な時間が経過し利用関係が収束した部品の集合に対して，各部品間に存在する利用関係に基づいてグラフおよび行列を構築し，構築された行列に対して繰り返し計算を行う事で各部品を評価する．求められる値は，開発者が利用関係に沿って参照を行うと仮定した場合の各部品の参照されやすさを表しており，よく利用される部品や，重要な部品から利用される部品の順位は高くなる．

2.1.1 で述べたように，実際の部品の集合には多数のコピーや類似した部品が存在している．異なるシステムをまたいで全く同じ，もしくはほとんど似た部品が現れる場合，それらの部品が再利用されたものと推測できる．そのため，CR 法では部品群を部品の単位としてみなす．それぞれの類似した部品への利用関係が一つの部品群への利用関係とみなされるため，コピーされた部品への評価を高くすることが可能となる．

2.2　システム構成

SPARS-J のシステム構成を図 1 に示す．

以下，図中の各項目について説明する．

- ライブラリ

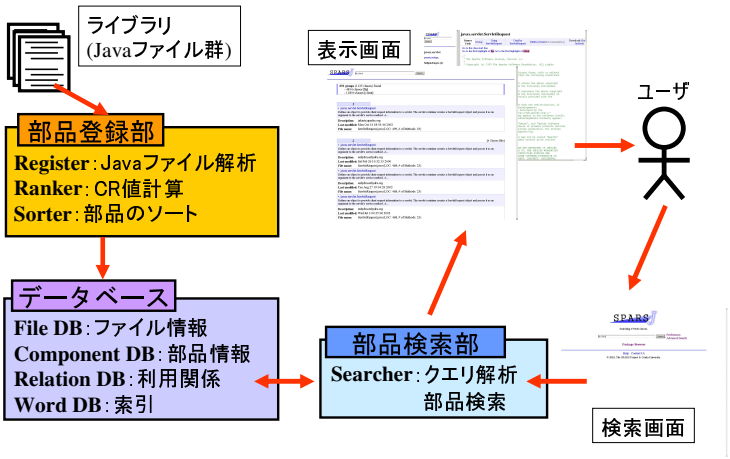

図 1: SPARS-J の構成

収集した Java ソースファイルが蓄積されているライブラリ．SPARS-J 上では部品の完全限定名で部品の分類を行なうので，ディレクトリ階層をパッケージに合わせる必要は特にない．検索結果の部品詳細表示では，ライブラリ中の該当部品が存在するファイルを参照し，表示する．

- データベース (File DB, Component DB, Relation DB, Word DB)
  ライブラリ中のファイル情報を管理する File DB，登録された部品に関する情報を管理する Component DB，部品間の利用関係を管理する Relation DB，索引の管理をする Word DB がある．各 DB は，ID からファイル名や部品名，ファイル名や部品名から ID を対応付けるために正引き，逆引き索引を持つ．

  – File DB
    ライブラリ中の Java ファイルのファイルパス，ファイル ID，各ファイルに存在する部品の ID，更新時間などの情報が格納されている．さらに，将来の拡張 (他言語対応など) に備えてファイルの記述言語情報も格納している．

  – Component DB
    部品名，部品 ID，部品が存在するファイル ID，ファイル中の部品の場所など部品とライブラリ中のファイルを File DB を介して対応付けるための情報の他に，各部品中で定義されたメソッドやフィールド名，類似判定のためのメトリクス，各部品の類似度，部品が属する部品群 ID，CR 値，部品中に現れた索引キー ID や出現頻度，場所など部品に関する全情報が格納されている．

  – Relation DB

部品間の利用関係と，その利用関係の種類が格納されている．その他，登録過程でまだ登録されていない部品への利用関係があった場合，利用する部品名を未解決名として格納し，部品が登録され次第利用関係を追加するための未解決名情報が格納されている．

- Word DB
  登録された部品中で出現した全索引キーとその ID を格納する DB である．検索時の大文字小文字の区別のために区別あり索引情報と，区別なし索引情報を格納している．

- 部品登録部 (Register, Ranker, Sorter)
  部品登録部はライブラリのファイルを解析して，各種情報をデータベースに格納する Register，利用関係に基づいて部品群の CR 値を計測する Ranker，検索時の高速化のために CR 値によって部品群を降順にソートし，各部品の索引キーとその出現頻度から索引キーの重みを計測する Sorter，以上 3 つのモジュールから構成される．

  - Register
    利用者が指定した Java ソースファイルを解析して，部品の切り出しの他，索引，メトリクス，利用関係などの抽出，さらに類似度の測定と部品群化を行なう．ファイル，部品，部品群，索引キーへの ID を割り振り，正引き，逆引き索引も作成する．サブモジュールとして，ソースファイル解析部 Parser と部品群化部 Cluster を持つ．
  - Ranker
    Register が抽出した利用関係から，各部品の CR 値を計測する．
  - Sorter
    SPARS-J は部品検索時に CR と KR の順位の統合を行なうため，データベース構築時にそのオーバーヘッドを軽減しておく必要がある．そのため，CR 値によって部品群を降順にソートしておき部品検索時には既に CR でソート済みのリストを取得できるようにする．また，KR 値は部品検索時に計測するが，Sorter 索引キーと出現頻度から索引キーの重みを計測しておくことで，検索時に KR 値の計測を高速に行なうことができる．

- クエリ解析部・部品検索部・データ表示部 (Searcher)
  Searcher はクエリの解析，部品の検索，データ表示を行なうモジュールである．利用者が指定したクエリを解析し，得られた検索キーをもとにデータベースの検索を行なう．得られた結果を順位付けし，各部品の KR 値の計測，CR と KR の順位の統合を

行なう．検索結果と併せて，部品の詳細情報として，ソースコードや利用関係，メソッド一覧などを利用者に対して提供する．

2.3 本研究との接点

SPARS-J では，ソースコード中の文字列を切り出した索引語や，クラス同士の関連を解析した利用関係など，登録されている部品そのものから取得した情報を基にして検索を行う．一方，利用者の検索履歴からは，同時に使用される部品や頻繁に使用される部品といった情報を取得する事ができる．現在，SPARS-J では検索履歴を利用していないが，検索履歴を利用する事により，検索者はより有用な検索結果を得る事ができると考えられる．

そこで，本研究では検索履歴から検索者に必要な部品を推測する，協調フィルタリングによる推薦手法を SPARS-J に対して適用する．

3 協調フィルタリング

3.1 関連研究

協調フィルタリング (Collaborative Filtering) の研究は，大量のアイテムの中から，個々のユーザの好みに合うアイテムを推薦する，推薦システム (Recommnedation System) の要素技術として研究が進んでいる． アイテムとは推薦の候補となるもの (記事，映画，音楽，あるいは，本研究ではソフトウェア部品) である．

協調フィルタリングに対する概念として，コンテンツベースフィルタリング (Content-based Filtering) がある．これはアイテム自体の性質，例えば含まれる文字列などを基にユーザにアイテムを提示する手法である．コンテンツベースフィルタリングの例としては，キーワード検索や，SPARS-J における利用関係の表示がある．

これに対し，協調フィルタリングにおいては他のユーザ評価を基にユーザにアイテムを提示する．初期の協調フィルタリングシステムの一つとして，Goldberg らの Tapestry[3] がある．これは，E メールおよび Netnews のフィルタリングシステムであり，ユーザは，キーワードと，評価しているユーザを指定して情報を得ていた．このシステムでは，推薦を互いに知り合っている人々に依存しており，大規模なコミュニティにおけるシステムにおいては適用できない．

一方，推薦相手や推薦する情報をシステムが自動的に決定するシステムとして Resnick ら [9] の GroupLens があげられる．GroupLens は，Usenet の記事を対象としたシステムであり，個々のユーザに対して興味を持つと思われる記事を推薦する．GroupLens では，ユーザは記事を読んだ後にその記事に対して 5 段階で評価し，システムに入力する．システムは，その評価に対して協調フィルタリング手法を適用し，記事をユーザに推薦するこの手法はユーザ間の相関を基に推薦値を計算する方法であり，相関係数法[1]とも呼ばれる [16].

Resnick らの方法では，アイテムに対する評価をユーザからの入力に頼っている．これに対して，オンラインショップの購入履歴や web のアクセス履歴などをユーザの好みの度合として解釈して利用するシステムも提案されている．前者は明示的な投票 (explicit vote) を利用するシステム，後者は暗黙的な投票 (implicit vote) を利用するシステムといわれる [8]

暗黙的な投票を用いるシステムとしては，Terven ら [7] の PHOAKS があげられる．PHOAKS は，URL を推薦するシステムである．PHOAKS は Usenet の記事から URL を抽出し，多く出現する URL を優先的に推薦する．

本研究は，暗黙的な投票を利用するシステムに分類される．利用者の部品の表示履歴を記録し，利用者は表示した部品に対して投票を行なったとみなし，協調フィルタリングに利用する．

[1]以降, 相関係数法と表記した時は原則として Resnick らの手法をさすものとする．

推薦のアルゴリズムは，多くの協調フィルタリングの研究の基礎となっている Resnick らの手法を元に, ソフトウェア部品の推薦に適した拡張を加えて利用する．

4 節での提案手法の説明に先だって，以降，相関係数法について説明する．

3.2 相関係数法

相関係数法の大まかな流れは以下の通りである．

1. 相関係数の計算
   対象ユーザ以外のユーザとの相関係数，つまり類似度を求める

2. 推薦値の計算
   それぞれアイテムについて，推薦値を計算する．推薦値は，相関係数を重みとして，各ユーザのアイテムに対する評価値の重み付き平均で求められる．

ここでは，ユーザ a に，アイテム k を推薦する事として説明する．

3.2.1 相関係数の計算

まず，ユーザ i のアイテム j に対する評価を $v_{i,j}$，ユーザ i が評価したアイテムの集合を I_i とし，ユーザ i の評価の平均値 $\bar{v_i}$ を次の式で求める．

$$\bar{v_i} = \frac{1}{|I_i|} \sum_{j \in I_i} v_{i,j} \tag{1}$$

次に，ユーザ a とユーザ i の相関係数 $c(a,i)$ を次式により，a 以外の全ユーザについて求める．この計算結果は [-1,1] の間に正規化される．

$$c(a,i) = \frac{\sum_{j \in I} (v_{a,j} - \bar{v_a})(v_{i,j} - \bar{v_i})}{\sqrt{\sum_{j \in I} (v_{a,j} - \bar{v_a})^2 \sum_{j \in I} (v_{i,j} - \bar{v_i})^2}} \tag{2}$$

ここで $I = I_a \cap I_i$ である．

3.2.2 推薦値の計算

ユーザ a に対するアイテム k の推薦値 $p_{a,k}$ は次式で求められる．ただし,U は k に対する評価を持つ全ユーザの集合を表す.

$$p_{a,k} = \bar{v_a} + \frac{\sum_{i \in U} c(a,i)(v_{i,k} - \bar{v_i})}{\sum_{i \in U} |c(a,i)|} \tag{3}$$

これはユーザ a との相関係数を重みとして全ユーザのアイテム k に対する評価の重み付き平均をとった物である．

4　提案手法

4.1　アルゴリズム

本研究では，3 節で説明した相関係数法を元にした手法によりソフトウェア部品の推薦を行なう．しかし，相関係数法は，本来ソフトウェア部品の推薦を前提としたアルゴリズムではない．そこで，以下に説明するような拡張を行なった物を，ソフトウェア部品のための推薦アルゴリズムとして提案する．まず，概念の対応付けについて述べたあと，相関および推薦値の計算について述べ，最後に計算例を示す．

4.1.1　概念の対応

ユーザ　相関係数法で求められるユーザ間の相関係数は，ユーザ同士の好みの類似度と言い替える事ができる．ユーザの好みは変化しない事が前提となっており，この前提は自然であると考えられる．これに対して，ソフトウェア部品検索において好みに対応する概念は検索目的であるが，これは常に同じであるとは考えにくい．そのため，利用者の過去の履歴全てを利用して推薦の計算を行なった場合は，正しい相関を求める事ができず推薦の精度が落ちる可能性が高いと考えられる．利用者の検索目的が変化した所で履歴を区切り，別ユーザとして扱う事が適切であるが，利用者からの明示的な入力は使用しないため，区切る箇所はシステム側で推測する事が必要である．検索目的の変化時として履歴を区切る方針としては以下の 3 通りが考えられる．

1. 時間の経過
2. 閲覧部品数
3. 検索システム使用の区切り

1,2 は, 利用者の環境, また検索目的により個人差が大きいと考えられ，妥当な区切りを見出す事が難しいと思われる．

3 は，検索者はある目的をもってシステムを使いはじめ，目的を終えた所で使用を終了するという仮定に基づいている．全ての検索者が該当するとは限らないが，妥当な区切りであると考えられる．

なお，この区切りは一般には検出が難しいが，SPARS-J のように WWW ブラウザ経由で利用するシステムでは，ブラウザウィンドウでシステムを表示してからそのウィンドウを閉じるまで (セッション) をいくつかの方法により把握する事ができる．よって，1 セッションを 1 ユーザとして扱う．

アイテム　アイテムは，部品に対応させる事が考えられるが，異なる類似部品への表示履歴があった場合には，それらの履歴は同じ部品への投票として扱う方が適切だと考えられる．部品を単位としてユーザ間の相関を計算すれば，実質的に同じ部品に対する表示履歴が別のものとして扱われ，正しい相関が求められないからである．そこで，類似部品群を単位として履歴を記録し，推薦を行なう．

ただし，以降の相関係数および推薦値の計算での説明においては，簡単のために，類似部品群ではなく単に部品として説明する．

投票　本研究では，ユーザの部品群に対する評価値として，ユーザ自身の入力などによる明示的な投票は利用しない．代わりにユーザの表示履歴を暗黙的な投票とみなして評価値とする．ユーザの部品群に対する評価値は，その部品群に対する表示履歴があれば 1，表示履歴がなければ 0 とする．

4.1.2　相関係数の計算

本研究では，1 セッションをユーザに対応させるため，1 ユーザあたりの評価部品数が少なくなると考えられる．これにより式 (2) での I の個数が少なくなる傾向にある．この場合，2 ユーザ間の相関係数を求めるための情報量が少なくなり，手法がうまく働かない．また，評価値は 0 か 1 かの 2 値であるため，求められる値は必ず 1 となり意味のある値とならない．Breese らは，これらの問題に対応する手法として，Default Voting という手法を提案している [6]．本研究では，この手法を用いて相関係数を求める．

この手法では，式 (2) において，I として対象のユーザ a，i のどちらかが評価した部品の集合およびどちらも評価していない部品 k 個の集合が計算に含められる．評価していないアイテムについては，規定の値 d を評価値として利用する．一般には d として，中立的な評価を表す値，もしくは低い評価を表す値が用いられる．本研究では $d = 0$ となる．また，$k = 10000$ とした．これは [6] の評価実験において用いられた値であり，本研究でも疑似的な履歴により適当であると判断し，決定した．

以上より，I_d を a,i のどちらも評価していない部品の集合 ($|I_d| = k$) とし，$I = I_a \cup I_i \cup I_d$ として，相関係数 $c(a, i)$ を求める式は次の様になる．

$$v'_{i,j} = \begin{cases} v_{i,j} & \text{if } j \in I_i \\ d & \text{if } j \notin I_i \end{cases} \tag{4}$$

$$\bar{v}_i = \frac{1}{|I|} \sum_{j \in I} v'_{i,j} \tag{5}$$

$$c(a,i) = \frac{\sum_{j \in I} (v'_{a,j} - \bar{v}_a)(v'_{i,j} - \bar{v}_i)}{\sqrt{\sum_{j \in I} (v'_{a,j} - \bar{v}_a)^2 \sum_{j \in I} (v'_{i,j} - \bar{v}_i)^2}} \tag{6}$$

なお，実際には計算量を考慮し，式 (6) は次式の様に変形して利用する．

$$c(a,i) = \frac{(n+k)(\sum_{j \in I} v'_{a,j} v'_{i,j}) - (\sum_{j \in I} v'_{a,j})(\sum_{j \in I} v'_{i,j})}{\sqrt{((n+k)(\sum_{j \in I} (v'_{a,j})^2) - (\sum_{j \in I} v'_{a,j})^2)((n+k)(\sum_{j \in I} (v_{i,j})^2) - (\sum_{j \in I} v_{i,j})^2)}} \tag{7}$$

ただし $I = I_a \cup I_i$,$n = |I|$ である．

4.1.3 推薦値の計算

相関係数法では，式 (3) により部品 j のユーザ a に対する推薦値を求めるが，その際の計算に含められるユーザの集合 U は，j に対して投票を行なったユーザの集合である．しかし，部品に対する評価値は，あるとすれば全て 1 であるため，求められる推薦値は必ず 1 となり，意味のある値とならない．これを避けるため，U は a との相関係数が 0 より大きく，かつ，表示した部品数が 2 部品以上であるような全てのユーザの集合とする．この時，部品 j に対する投票の無いユーザは，相関係数を求めた時と同様に，評価値 0 で投票しているとして扱う．

ここで相関係数が負のユーザを U から除外したのは，ソフトウェア部品検索において負の相関は意味をもたないと考えられるからである．記事や映画といった，個人の趣向が反映される分野ならば「好みが正反対」，すなわち自分の好む物を相手が嫌い，自分の嫌う物を相手が好むという事はありうる．しかしソフトウェア部品に関しては，相関が負でもそのユーザの必要する部品が自分にとって必要がないとは限らないと考えられる．また，相関が求められているという事から，最低 1 部品は履歴が重複していて幾らかの関連性があるため，負の相関として扱う事は正しくないと考えられる．

また，表示部品数が 1 部品のみであるユーザも除外したのは，明らかに推薦を求めることには役立たない上に，推薦値の計算時に全て 0 を投票している扱いとなるため，推薦値を全体的に下げる働きをするためである．

さらに，相関係数 $c(a,i)$ に対し,Case Amplification[6] を適用する．これは，式 (3) において $c(a,i)$ の代わりに， $c(a,i)$ を ρ 乗した値 $c'(a,i)$ を使用する手法であり，次式で表される．

$$c'(a,i) = \begin{cases} c(a,i)^\rho & \text{if } c(a,i) \geq 0 \\ -(-c(a,i)^\rho) & \text{if } c(a,i) < 0 \end{cases} \tag{8}$$

Case Amplification により，相関の高いユーザはより大きい重みが加わり，相関の低いユーザはより低い重みが加わるようになる．本研究では疑似的な履歴による試行により，$\rho = 2$ とした．

また，ユーザの評価値の平均値は全てのユーザに関して 1 であるため推薦値の計算において評価値の平均値は考慮しない．

以上より，ユーザ a に対するアイテム k の推薦値は次式によって求められる．

$$p_{a,k} = \frac{\sum_{i \in U} c'(a,i) v'_{i,k}}{\sum_{i \in U} |c'(a,i)|} \tag{9}$$

ただし，U は，$c(a,i) > 0$ かつ $|I_i| > 1$ であるユーザ i の集合である．

4.1.4 計算例

提案手法での推薦値の計算例を，表 1 で示される履歴の中のセッション 5 のユーザに対して示す．ここでは簡単のために部品群については考慮せず，部品に対する推薦を計算する．表中の “◯” は，セッション中で部品が表示された事を表す．

まず， セッション 5 とセッション 1 の相関係数 $c(5,1)$ を求める．

$$\begin{aligned} c(5,1) &= \frac{(4+10000)(0\cdot 1+1\cdot 1+1\cdot 1+0\cdot 1)-(0+1+1+0)(1+1+1+1)}{\sqrt{((4+10000)(0^2+1^2+1^2+0^2)-(0+1+1+0)^2)(4+10000)(1^2+1^2+1^2+1^2)-(1+1+1+1)^2))}} \\ &= 0.71 \end{aligned}$$

同様にしてセッション 2,3,4 に対しても求めると，表 2 の様になる．

続いて，部品 1 に対する推薦値を求める．セッション 5 との相関係数が 0 より大きいセッションは 1,2,4 であり，推薦値は以下の様になる．

$$\begin{aligned} p_{5,1} &= \frac{0.71^2\cdot 1+0.71^2\cdot 1+0.32^2\cdot 0}{|0.71^2+0.71^2+0.32^2|} \\ &= 0.91 \end{aligned}$$

同様にして他の部品に対して求めた推薦値を表 3 に示す．

4.2 SPARS-J への実装

4.2.1 概要

本研究では，SPARS-J に対して以下の機能を追加した．

- 推薦機能の利用/不利用の選択機能
  本機能は，Cookie を用いて利用者を識別し，履歴をデータベースに記録する． このように自らの履歴を記録される事を好まない利用者に配慮し，SPARS-J の Preferences(図 2) にて推薦機能を利用するかどうかを選択できる． 推薦機能を利用しない時は，履歴の記録・推薦の表示のどちらも行なわない．

表 1: 例:表示履歴

セッション＼部品	1	2	3	4	5	6	7	8	9	10
1	○	○	○	○						
2	○	○	○		○					
3						○		○		○
4			○			○	○	○	○	
5		○	○							

表 2: 例:セッション 5 との相関係数

セッション	相関係数
1	0.71
2	0.71
3	0
4	0.32

表 3: 例:セッション 5 に対する各部品の推薦値

部品	1	2	3	4	5	6	7	8	9	10
推薦値	0.91	(表示済み)	(表示済み)	0.45	0.45	0.09	0.09	0.09	0.09	0

- 履歴取得機能
  利用者の各セッションに対して一意となる ID を与え，表示したソースコードの履歴をデータベースに保存する．

- 履歴閲覧機能
  利用者がセッション開始から現在までに表示した部品を一覧する．それぞれ，部品のソースコードを表示するページへのリンクとなっており，ここからその部品の情報を表示する事も可能である．

- 履歴削除機能
  履歴一覧の中から，適合していない部品を削除する．これにより，そのセッションに対する推薦の精度を向上させるとともに，以降の利用者のセッションに対しても推薦の精度向上が期待できる．

- 推薦部品表示機能
  4.1 で説明したアルゴリズムにより作成された推薦部品の一覧を表示する．各部品の文字列は，その部品の情報を表示するページへのリンクとする．また，推薦値も併記する事により，利用者はその部品の適合性をどの程度期待できるかを知る事ができる．推薦部品は 2 通りの方法で表示する事ができ，利用者は目的に応じて使い分ける事ができる．

  – 全推薦部品の表示
    推薦部品を，推薦値順に全て表示する．
  – 利用関係にある推薦部品の表示
    現在表示している部品と利用関係にある部品の一覧を推薦部品で絞りこみ，推薦値順に表示する．これは SPARS-J の利用関係の表示機能と同様に，利用関係，非利用関係を選択して表示する事ができる．

4.2.2 履歴取得の実装

セッションの識別 利用者のセッションの識別は，セッション Cookie を利用して行なう．Cookie とは， Web ページから送信した情報をブラウザ側に保存する機構であり，セッション Cookie は，ブラウザを終了した時に削除される．このセッション Cookie にセッションを識別するための ID を保存し，履歴記録時に取得して利用する．もしこの Cookie がなければ，セッションの最初の部品表示であると判断して ID を新たに発行し， 履歴格納に利用すると同時に Cookie に保存する．

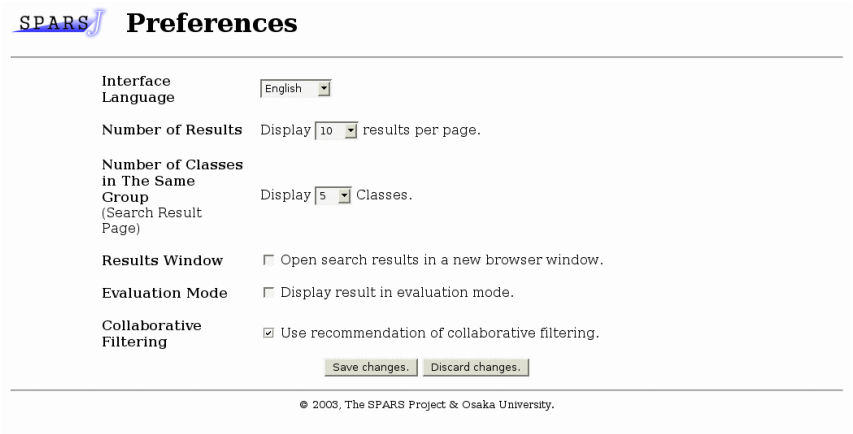

図 2: SPARS-J の Preferences 画面

履歴情報　履歴としてデータベースに保存する内容は以下の通りである．

- セッション ID
  前述した，利用者のセッションを識別する ID である．
- コンポーネント ID
  表示した部品のコンポーネント ID である．推薦値の計算そのものには利用しないが，推薦部品の表示時に，同一類似部品群の中から，優先して表示する部品を選ぶために使用する．
- グループ ID
  表示した部品の類似部品群の ID である．グループ ID は，コンポーネント ID から一意に求める事ができるため推薦の計算時にも取得する事ができるが，計算の効率のために履歴として保存する．
- 削除フラグ
  履歴保存時は 0(false) の値を入れる．後で述べる履歴削除機能を利用した時にこの値は 1 となり，推薦の計算時に無視されるようになる．

4.2.3　推薦部品表示部の実装

本機能を利用するための画面を図 3 に示す．SPARS-J では，部品表示部において，ソースコード，利用関係などの表示内容の切り替えにタブ型のインターフェースを利用している．そのタブの並びに “Recommendation”(“推薦”) を追加し，そのリンクをクリックする事で本機能の利用を開始できる．

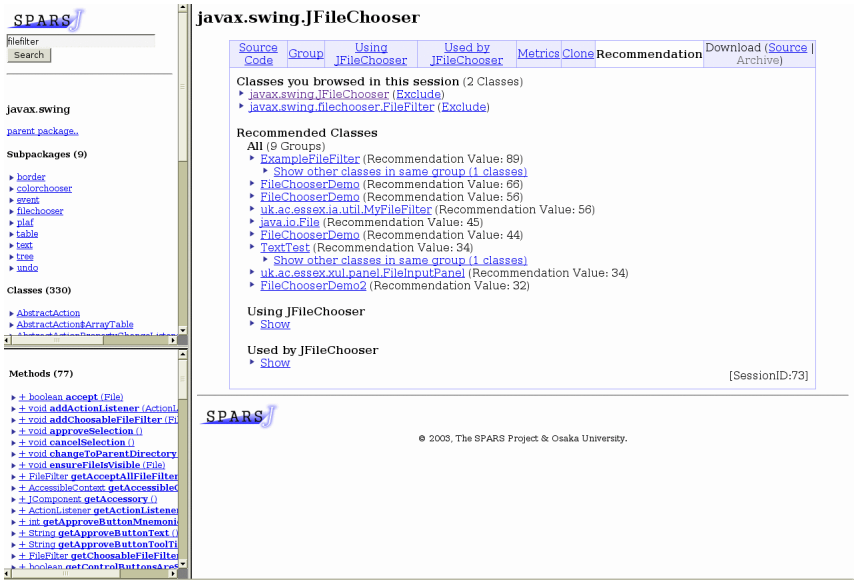

図 3: 推薦機能の画面

図 4: 推薦機能の画面 (部品群の表示)

図 3 において，上から順に表示履歴一覧，推薦部品一覧となっており，推薦部品一覧は，リンクにより全ての推薦部品，利用関係にある推薦部品，被利用関係にある推薦部品を切り替える事ができる．初期状態では表示履歴と，全ての推薦部品が表示されている．

以下，それぞれの項目について詳しく述べる．

- 表示履歴

  Cookie から取得したセッション ID をキーとして，データベースから該当する履歴を取得し，部品の重複を除いた上で一覧表示する．この時，削除フラグが 1 となっている物は無視する．

  履歴の削除は，部品名の横の “Exclude”(“除外”) というリンクをクリックする事で行なう．削除は，セッション ID と部品のコンポーネント ID をキーとして履歴を取得し，削除フラグを 1 として保存する．

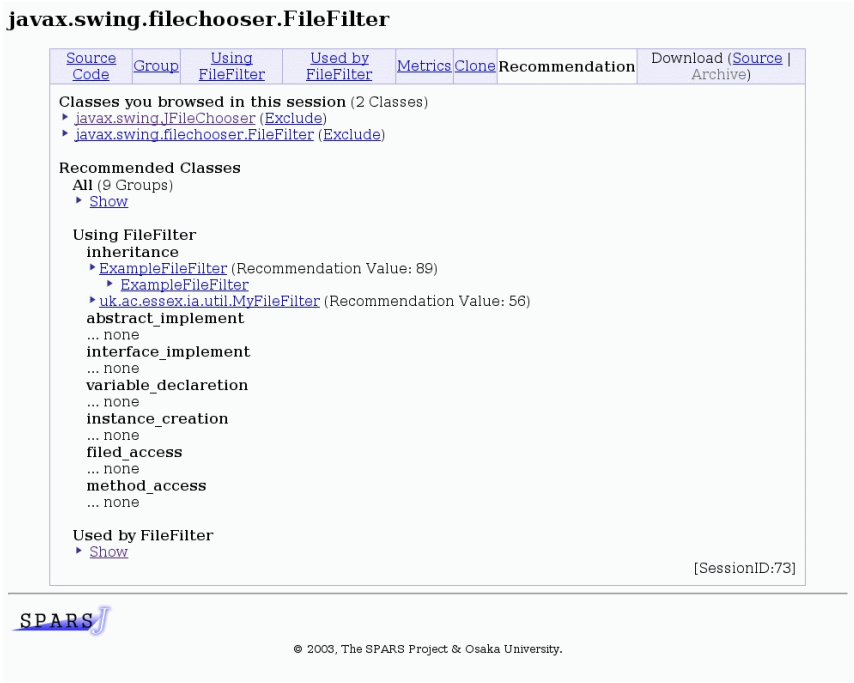

図 5: 推薦機能の画面 (利用関係の絞り込み)

- 推薦部品

  4.1 で説明したアルゴリズムによって，推薦部品の一覧を得る．得られる推薦は類似部品群を単位としているために，部品群中の部品を表示する事になるが，全て表示すると部品数が多い場合に推薦部品が見づらくなるという問題がある．そこで，図 4 に示すように 1 部品のみを表示して，同一部品群中の他の部品は表示するためのリンクをクリックした時に表示されるようにする. なお，この時に優先して表示される部品は，過去の表示履歴中に存在する部品とする．

  – 全て

    4.1 で説明したアルゴリズムにより推薦値が求められた部品のうち，閾値 t 以上の部品を推薦値順にソートして推薦値とともに表示する．本システムでは，$t = 0.3$ とした．この値は実験的に求めた物であるが，部品群および利用方法によって調整が必要であると考える．

    また，既に表示済の部品については，利用者はその適合性について既知であると考え，推薦部品には含めない．

  – 利用関係にある部品

    表示中の部品と利用関係にある部品を，利用関係の種類ごとに，上述した「全て」

の推薦部品群で絞り込み，推薦値順にソートして推薦値とともに表示する．

利用関係にある推薦部品を表示している様子を図 5 に示す．

– 被利用関係にある部品

利用関係にある部品と同様，表示中の部品と利用関係にある部品を，利用関係の種類ごとに,「全て」の推薦部品群で絞り込み，推薦値順にソートして推薦値とともに表示する．

5　適用実験

5.1　実験概要

実験の目的は，協調フィルタリングによる推薦機能の，ソフトウェア部品検索における有効性を確認する事である．

実験内容は，課題として用意した Java のスケルトンコードの未実装部分を実装する事である．この時，被験者は課題と同時に渡した参考資料および SPARS-J のみを参照してプログラム作成を行なう．

以下，具体的な課題プログラムの内容，被験者，課題の作成環境について述べる．

課題プログラム　課題は，練習課題 1 種類を含む 5 種類用意した．それぞれ，実装する処理以外は全て記述済みであり，該当部分を記述してコンパイル・実行すれば動作が確認できる様なプログラムとした．また，各課題の内容はそれぞれ独立しており，課題プログラムが他の課題の答えやヒントを含むような事は無い．

練習課題を P0，課題をそれぞれ P1，P2，P3，P4 とすると，詳細は以下の通りである．

P0 ZIP 書庫に関する課題
　指定されたファイルを ZIP 書庫に圧縮し，また展開するプログラムを作成する．

P1 画像処理に関する課題
　指定されたファイルを読み込み，別形式で保存するプログラムを作成する．

P2 FTP に関する課題
　指定されたサーバーに接続・ログインし，サーバー上のファイルを取得するプログラムを作成する．

P3 Java の GUI ライブラリ Swing に関する課題
　ファイル選択ダイアログを開き，ユーザが選択したファイルのファイルパスを取得するプログラムを作成する．

P4 JDBC に関する課題
　指定されたデータベースに接続・ログインし，データベースから情報を取得するプログラムを作成する．

被験者　実験は，大阪大学大学院情報科学研究科の学生および研究員，合計 8 名に対して行なった．各被験者は学部 3 年次の演習，もしくは各々の研究で対象言語である Java を扱っ

ており，言語に対する知識はもっている．また，事前に，各々に対して，SPARS-J の利用方法および，各課題に必要な前提知識の講習を行なった．

被験者は，実験時には 2 つのグループ GP1,GP2 に分けた．

課題の作成環境　被験者が課題作成時に利用する SPARS-J のデータベースは，以下の様な内容で開始した．

- 検索対象部品
  実験で利用する SPARS-J に登録した部品は以下の通りであり，総クラス数は約 35,000 クラスである．この内容のみで課題プログラムが全て作成できる事を確認し，実験を行なった．
  - JDK1.4 のソースおよびデモコード
  - The Jakarta Project[13] で開発されているアプリケーションおよびライブラリのソースコード
  - Apache Ant[12] のソースコード
  - SourceForge 上で開発されているアプリケーションのソースコード
  - Eclipse[4] のソースコードおよびデモコード
  - その他，インターネット上から収集したコード
- 表示履歴
  実験開始時は，ユーザの表示履歴は全く無い状態とした．

5.2　実験手順

1. まず，練習段階として被験者全員が P0 を行なう．これは，SPARS-J の利用，および課題に十分に慣れ，後の Exp1，Exp2 間で慣れによる差異が発生しないようにするためである．これは推薦機能を利用しない．

   また，この時に作業時間，検索効率などを測定し，その結果を元に被験者の能力が平均的になるようにグループ分けを行なう．

2. (Exp1) GP1 の被験者が課題 P1，P2 を，GP2 の被験者が課題 P3，P4 を，それぞれ推薦機能を使用せずに作成する．

3. それぞれの被験者のプログラムを参照し，客観的に適合していない部品の履歴を削除する． これは，履歴の削除機能を使用したとみなされる．

4. (Exp2) GP1 の被験者が課題 P3，P4 を，GP2 の被験者が課題 P1，P2 を，それぞれ推薦機能を使用して作成する．

   ここで，Exp2 の進行によってもユーザの履歴は蓄積されるが，Exp2 開始以降の履歴は，他の被験者の推薦に使用されないようにシステムに改造を加えて実験を行なった．

5.3 実験結果

実験結果は表 4 の様になった．表中の A1,A2,A3,...,A8 はそれぞれ被験者を表す．
以下，表中の項目について説明する．

- 検索時間
  被験者が，課題開始から終了までにかかった時間のうち，プログラムをコーディングしていた時間を除いた時間である．この時間が短いほど，検索効率が良いと言える．

- 表示部品数
  検索中に表示した部品の個数である．履歴削除機能により削除された部品も含まれている．

- 適合部品数
  実際に被験者が利用したか否かに関わらず，部品中のコードを課題プログラムに使用したとみなせる部品，および部品自体を課題プログラムに使用しているような部品の個数である．

- 適合率
  適合部品数を表示部品数で割った値である． この値が 1 に近いほど，検索効率が良いと言える．

5.4 分析と考察

実験結果から，各課題とも，推薦機能を使用したグループの方が平均で作業時間・適合率ともに優れている事が確認できる．

結果より，課題 P3 では推薦機能の有無による差異が小さい事が分かる．これは，推薦機能無しで課題プログラムを作成したグループ GP2 中の被験者 A5・A6 が，課題の分野である GUI プログラミングの経験者であったために，部品を推薦されなくても，検索結果や利用関係の一覧から適合するような部品を判別し，利用することができたためであると推測される．逆に，どちらのグループにも該当分野の経験者が居なかった課題 P1 では大きな差が

表 4: 実験結果

		GP1 (推薦機能 無し)					GP2 (推薦機能 有り)				
		A1	A2	A3	A4	平均	A5	A6	A7	A8	平均
検索時間 (分)	P1	28	47	14	50	34.8	4	3	28	15	12.5
	P2	25	19	2	28	18.5	2	4	5	2	3.2
表示部品数	P1	19	15	13	19	16.5	3	3	9	12	6.8
	P2	17	8	1	20	11.5	2	2	5	2	2.75
適合部品数	P1	8	8	3	5	6	3	3	7	9	5.5
	P2	1	3	1	3	2	2	2	2	2	2
適合率	P1	0.42	0.53	0.23	0.26	0.36	1	1	0.78	0.75	0.89
	P2	0.06	0.38	1	0.15	0.18	1	1	0.4	1	0.73

		GP2 (推薦機能 無し)					GP1 (推薦機能 有り)				
		A5	A6	A7	A8	平均	A1	A2	A3	A4	平均
検索時間 (分)	P3	9	18	44	26	24.2	14	60	3	9	21.5
	P4	23	45	13	23	26	3	38	7	12	15
表示部品数	P3	3	7	18	19	11.8	9	26	3	3	10.2
	P4	11	16	11	11	11.25	4	11	5	5	6.3
適合部品数	P3	3	7	10	7	6.8	9	13	3	3	7
	P4	6	10	8	8	8	4	11	5	5	6.2
適合率	P3	1	1	0.55	0.37	0.73	1	0.5	1	1	0.88
	P4	0.55	0.63	0.73	0.73	0.66	0.67	0.64	0.83	1	0.79

表 5: アンケート結果

	A1	A2	A3	A4	A5	A6	A7	A8	最頻値
推薦部品が役立ったか	4	3	5	2	5	4	4	5	4,5
UI は使いやすかったか	3	3	4	3	3	5	3	3	3

でている．これは，推薦機能が特に検索目的が利用者にとって未知の分野である場合に検索効率の向上に役立つ事を示唆している．

各被験者毎に見た時は，被験者 A2,A7 を除く全ての被験者で推薦機能を利用した時の方が作業時間・適合率が優れている．ここで，被験者 A2 については作業時間の面で,A7 については適合率の面で，推薦機能の使用により良くなったとはいえない．これは以下の理由による物だと考えられる．

被験者 A6 の検索方法は，推薦部品を含む多くの部品に軽く目を通した後で必要な部品を探すという方法であった．そのため，推薦部品がその他の部品と一緒に表示履歴に入ってしまい，有効に活用されなかったと考えられる．

この事から，表示履歴に含まれる部品も，利用者の操作で推薦部品に含めるかどうか選択できるようなインターフェースにすべきであると考えられる．

5.5 アンケートによる評価

実験終了後，全ての被験者に対して推薦機能に関する簡単なアンケートを行い，利用者視点での定性的な評価を行った．アンケート内容は以下の通りである．

1. 推薦された部品は役立ったかどうか.
   解答は 1 から 5 までのの 5 段階評価で，5 が一番良い評価とした.

2. ユーザーインターフェースは使いやすかったかどうか.
   1 と同様，5 段階評価とした.

3. その他, 評価できる点や改善点など
   自由記入とした.

このアンケートのうち，1 および 2 の結果を表 5 に示す．また，3 での意見をまとめたものを以下に示す.

- 評価できる点
  - 推薦部品が自分の表示履歴と一緒に表示されるため，推薦を自然に利用できた
  - 使用した事の無いライブラリやクラス群を利用する時に，一つのクラスからクラスの塊が得られるので便利
- 改善点
  - 推薦機能は，専用のページだけでなく他のページからも利用できた方が良い
  - 推薦クラスがクラス名のみでの提示だったので，適合性が判定しづらい

表5より，被験者に対して推薦された部品は概ね役立っていたという事がわかる．実験結果で示された検索効率の向上は，推薦機能によるものである事を裏付けていると考えられる．

これに対して，ユーザーインターフェースに関しては悪い評価を与えた被験者は居ないものの，優れていると評価した被験者も小数であった．また，自由記入欄での意見も合わせると，現在の実装は難があると言う程のものではないが，いくらかの改善の余地がある事がわかる．

6　まとめと今後の課題

本研究では，協調フィルタリングを利用したソフトウェア部品推薦手法を提案し，Java ソフトウェア部品検索システム SPARS-J に対して実装を行った．また，適用実験を行い，協調フィルタリングによる推薦を利用する事で，実際に検索効率が向上する事を確認した．特に，利用者が知識をもたない分野の検索においての有効性が示された．

今後の課題としては，

- ユーザーインターフェースの改良
  現在，推薦機能の実装は 1 ページにまとめられており，利用のためにはわざわざ見に行く必要がある．検索結果表示や，通常の利用関係の表示時にも，推薦を利用できるようにすべきだと考えられる．また，既に表示済みの部品についても推薦を行なうかどうか，利用者が指定できるようにするべきである．

- 履歴に対する重み付け
  ユーザの表示履歴の中には，当然適合している部品もあれば適合していない部品も含まれている． そのため，表示した部品は全て評価値 1 をつけるのではなく，履歴から適合性を推測して幅のある値をつけるべきである．例えば，部品 A を表示し，続いて「部品 A を利用している部品一覧」から部品 B を表示した時，部品 A は適合していると推測し，高い評価値をつける．

- より大規模な評価実験
  本研究での実験では，被験者の人数，履歴の量がともに少く，提案手法についての有効性が十分に確認されたとはいえない．よって，十分な履歴を蓄積した上で，多くの被験者に対して評価実験を行う必要がある．

が挙げられる．

謝辞

本研究において，常に適切な御指導および御助言を頂きました 大阪大学大学院情報科学研究科コンピュータサイエンス専攻　井上克郎教授 に深く感謝致します.

本研究において，常に適切な御指導および御助言を頂きました 同　楠本真二助教授に深く感謝致します.

本研究において，常に適切な御指導および御助言を頂きました 同　松下　誠助手 に深く感謝致します.

本研究において，常に適切な御指導および御助言を頂きました 独立行政法人 科学技術振興機構 計算科学技術研究員 山本哲男 氏 に深く感謝致します.

本研究において，常に適切な御指導および御助言を頂きました 大阪大学大学院情報科学研究科コンピュータサイエンス専攻 産学官連携研究員 横森励士 氏 に深く感謝致します.

最後に，その他様々な御指導，御助言を頂き，適用実験へ協力して頂きました 大阪大学大学院情報科学研究科コンピュータサイエンス専攻 ソフトウェア工学講座 井上研究室の皆様に深く感謝いたします.

参考文献

[1] B. Keepence and M. Mannion: "Using patterns to model variability in product families", *IEEE Software*, Vol. 16, No. 4, pp. 102-108, 1999.

[2] C. Braun: Reuse, in John J. Marciniak, editor, *Encyclopedia of Software Engineering*, Vol. 2, John Wiley & Sons, pp. 1055-1069, 1994.

[3] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry: "Using Collaborative Filtering to Weave an Information tapestry.", *Communications of the ACM*, Vol.35, No.12, pp.61-70, 1992

[4] Eclipse : "http://www.eclipse.org/".

[5] I. Jacobson, M. Griss and P. Johnsson : "Software Reuse", *Addison Wesley*, 1997

[6] J. S. Breese, D. Heckerman and C. Kadie: "Empirical Analysis of Predictive Algorithms for Collaborative Filtering", *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence*, pp.43-52, 1998.

[7] L. Terveen, W. Hill, D. McDonald, and J. Creter: "PHOAKS: A System for Sharing Recommendations", *Communications of the ACM*, Vol.40, No.3, pp.59-62, 1997.

[8] P. Resnick and H. R. Varian: "Recommender Systems", *Communications of the ACM*, Vol.40, No.3, pp.56-58, 1997.

[9] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl: "GroupLens: An Open Architecture for Collaborative Filtering of Netnews", *Proceedings of ACM 1994 Converence on Computer Supported Cooperative Work(CWSW'94)*, pp.175-186, 1994

[10] S. Isoda: "Experience report on a software reuse project: Its structure, activities, and statistical results", *Proceedings of 14th International Conference on Software Engineering (ICSE14)*, pp.320-326, Melbourne, Australia, 1992.

[11] SourceForge: "http://sourceforge.net/".

[12] The Apache Ant Project: "http://ant.apache.org/".

[13] The Jakarta Project: "http://jakarta.apache.org/".

[14] V. R. Basili, G. Caldiera, F. McGarry, R. Pajerski, G. Page and S. Waligora: “The software engineering laboratory - an operational software experience”, *Proceedings of 14th International Conference on Software Engineering (ICSE14)*, pp. 370-381, Melbourne, Australia, 1992.

[15] 北, 津田, 獅々堀: 情報検索アルゴリズム, 共立出版, 2002

[16] 山西: “Web マイニングと情報論的学習理論”, 情報学シンポジウム 2002, pp. 9-16, 2002.

[17] 横森, 藤原, 山本, 松下, 楠本, 井上: 利用実績に基づくソフトウェア部品重要度評価システム, 電子情報通信学会論文誌, D-I, Vol. J86-D-I, No.9, pp.671-681, 2003, and Technical Report of SE Lab, Dept. of Computer Science, Osaka University, SEL-Nov-21-2002, Nov. 2002