関係データベースを使った事例ベース検索 (1) − アルゴリズム*

6H−5

島津 秀雄　柴田 晃宏　高島 洋典†

日本電気 (株) C&C 情報研究所‡

1 はじめに

本稿では、関係データベースシステムを使って事例ベースを構築し、利用者の類似事例検索要求を複数の関係データベースの検索式に展開して nearest neighbor(最近隣) 検索を実現する手法について述べる。

我々は、既に事例ベース検索シェル CARET [1][2] およびその上のアプリケーションである SWQC 検索システム SQUAD [4][3] を開発した。そこでは、事例ベースは CARET 自身がもつ事例記憶部に格納されていた。これは、他の事例ベースの研究や商用化されたツール群でも同様である。しかしながら、この方法では以下の点で問題があることがわかった。

- 事例ベースプロジェクトは、既存のデータベースをより有効活用しようという場合が多い。その場合、従来のデータベースと別に事例ベースを新たに作成するのは無駄であるし、保守も困難であり、実現性が低い。
- 開発プロジェクトが全く新規に事例ベースを作成する場合でも、事例ベースの開発・管理・検索の処理は、従来のデータベースシステムのそれと重なる部分も多く、両者は全く合いいれないものではない。また、既存データベース管理システムが有する検索高速化の種々の技術は、事例ベース検索に於いても有効に使われるべきである。
- 我々の経験によると、事例ベースの検索システムを構築しようとして始めたプロジェクトにおいてすら、利用者は、いつも類似検索をしたいと望んでいるわけではなく、しばしば一般的なデータベース検索をしたくなる。
- 事例ベースを新規に独自に作成すると、事例ベース検索・推論のプロジェクトが失敗したり終了すると、せっかく作成した事例ベースもクズになってしまう。もし、商用データベース管理システムを使って事例ベースを構築しておけば、たとえ事例ベースプロジェクトが終焉しても、少なくともデータベースは残るというリスクヘッジの効果もある。

以上のことを鑑みると、事例ベースシステムを構築する場合、標準データベース構築システムを使って事例ベースを構築し、利用者は標準的なデータベース検索と類似事例検索の両方が出来ることが望ましい。そこで、我々は関係データベースシステムを使って事例ベースシステムを構築するツールとして CARET の新しい実装を行った。2節では、従来手法について述べる。3節では、CARET における類似事例検索方法のあらましを述べる。4節では、CARET の nearest neighbor 実現手法を説明する。

2 従来手法

従来の事例ベース推論においては、nearest neighbor 検索は、次のように実現されていた。ここで、個々の事例は、いくつかの属性の値の集合で表現され、事例ベースに格納されているとする。

1. 問題がいくつかの属性の値の集合の形で与えられる。

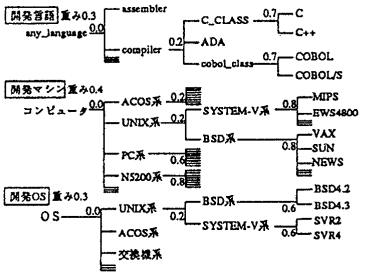

図 1: 属性内類似度定義の例

2. 事例ベースから事例を1つ取り出し、与えられた問題と事例の対応する属性の値同士の類似度を計算する。
3. 個々の属性の類似度に重みを掛けて加重平均した値を、その事例と問題との総合類似度として出力する。総合類似度の式は、以下のようになる。

$$\frac{\sum_{i=1}^{n} \text{属性 i の重み} \times \text{属性 i の問題と事例の類似度}}{\sum_{i=1}^{n} \text{属性 i の重み}}$$

4. 2,3 の処理を事例ベース中のすべての事例に関して行い、総合類似度の値の大きいものを与えられた問題に対する類似事例として出力する。

事例を表現する個々の属性には、それぞれ2つの属性値間の類似度が定義されている。例えば、事例内の1つの属性として「開発言語」があれば、その類似度は、図1のように意味の木構造で記述される。具体的な属性値は葉ノードに存在し、それらの共通概念が中間ノードの形で階層的に定義されている。個々のノードには、2つの属性値がそのノードで一致したときの類似度が記述されている。 2つの属性値が与えられると、木構造を上にたどって行き、両者の共通のノードを見つけ、そのノードに記述されている類似度が、与えられた2つの属性値間の類似度となる。例えば、2つの属性値が「C」と「C++」のときは、両者が「C-CLASS」下にあるので、類似度 0.7 となる。

3 CARET における類似事例検索手法

関係データベース上に事例ベース検索システムを構築する場合、データベースから1つずつレコードを検索してきて問題と比較する既存の方法で nearest neighbor 検索を実現するのは非効率である。

CARET においては、利用者から類似事例検索要求が与えられると、その要求を解釈して、それに対応する複数のレコード検索式 (SELECT 文) を作成し、それら検索式を順々に実行する、という手順になる。それら検索式の実行結果が利用者の検索要求に対する類似事例となる。今、個々の事例は、関係データベースの1レコードとして表現されているとする。利用者から与えられる問題は、やはりいくつかの属性の値の集合で

*Case-base Retrieval Using Standard Relatinal Database (1) − Algorithm

†Hideo SHIMAZU, Akihiro SHIBATA, and Yosuke TAKASHIMA

‡C&C Information Technology Research Labs., NEC Corporation

与えられる。CARETは、与えられた問題表現に対応する条件記述を含む検索式(SELECT文)を作成する。また、この検索式で検索されるであろうレコードの類似度を1(完全一致)とする。次に、問題の条件制約を少し緩めた条件記述を作成し、それを含むレコード検索式を作成し、この緩めた条件記述で検索されるであろうレコードの類似度を計算する。さらに、問題の条件制約を一層緩めた条件記述を作成し、同様な処理を行なう。次に、作成されたレコード検索式を類似度の高い順にデータベースに発行し、それから検索されたレコードの内容と対応する類似度の値を組にして、利用者に提示していく。

従来手法では、問題と個別の事例を比較して類似度を計算していったが、CARETでは、問題と類似な事例を得るための検索式とその検索式で事例レコードが得られた場合のその事例の類似度の値の計算を先に行ない、次にその検索式を使ってデータベース検索を行い、その結果として個別事例を手に入れる、というように処理の順序がちょうど逆転している。

4 第n次類似集合と総合類似集合

CARETによる類似検索手法では、与えられた利用者の条件に対して、利用者の条件制約を少しずつ緩めていくときの、条件記述自動生成およびその緩めた条件で検索した結果得られるレコードの類似度の値計算方法が重要である。ここではそれらを説明する。条件式の生成に使われる個別属性の属性内類似度定義の情報は、従来手法の場合と全く同じである。

1. 問題がいくつかの属性の値の組の形で与えられる。

2. 利用者から提示された個別の属性ごとに、図1のような属性内類似度定義を参照して、利用者から与えられた問題中の属性値に対して最も高い類似度を取る属性値の集合（第1次類似集合と呼ぶ）、2番目に高い類似度を取る属性値の集合（第2次類似集合と呼ぶ）、3番目に高い類似度を取る属性値の集合（第3次類似集合と呼ぶ）、のように、類似度の高い順に、属性値の集合を作っていく。最も高い類似度を取る属性値の集合を求めるには、属性内類似度定義において、利用者から与えられた属性値の値から出発してその親のノードを見つけ、その親の（自分自身を除く）子供の集合を作成すれば良い。また、親のノードに記述された類似度の値が、その類似度となる。第2次類似集合の場合には、親の親までたどり、第3次以降は更に祖先をたどって同様の処理をすればよい。例を示すと、図1の「開発言語」で、「C」が利用者から与えられた値だとすると、第1次類似集合は「C」で類似度1、第2次類似集合は「C++」で類似度0.7、第3次類似集合は「ADA, COBOL, COBOL/S」で類似度0.2、等となる。このように生成される類似集合は、与えられた入力属性値に依存して決まる。

3. 総合類似集合およびその総合類似度を生成する。属性単位に作成された第n次類似集合の1つを取り出して、属性の数の次元を持つ総当たり的組み合せをすべて作る。生成される組合せ数の上限は、$\prod_i^{(\text{指定属性数})}(\text{属性}i\text{の階層数})$であるが、すべての組合せを生成する必要はなく、類似度が大きいものに限定すれば組合せの数は減る。生成された個々の組み合せは、総合類似集合と呼ぶ。それぞれの総合類似集合の総合類似度は、各組合せ内の属性の第n次類似集合の類似度にその属性の重みを掛けて加重平均したものとして算出される。式で表わすと

$$\frac{\sum_{i=1}^{n} \text{属性i の重み} \times \text{属性i の第n次類似集合の類似度}}{\sum_{i=1}^{n} \text{属性i の重み}}$$

となる。例えば、図2のように、利用者が「開発言語」と「開発マシン」の2つの属性の値をそれぞれ「C」「EWS4800」と提示する形で問題を与えたときには、「開発言語」「開発マシン」の両方とも、それぞれ第3次類似度集合まで作成され、次にそれらの全組み合せが計9つ生成される。その組み合せの1つの例が、(「C」, [MIPS, ...])であり、その総合類似度は、$(1\times0.3+0.8\times0.4)/(0.3+0.4)=0.89$となる。

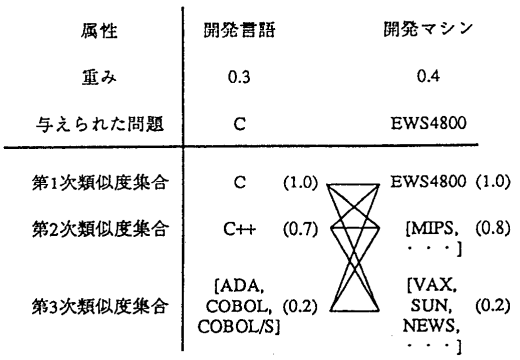

図 2: 総合類似度集合の生成

4. 総合類似集合からSELECT文の条件式(WHERE句)を生成する。これは単純に行なえる。

上の処理の流れの説明では述べなかったが、総合類似度の下限をしきい値として決めておけば、しきい値以下の類似度のSQL式の実行を抑制できる。また、上位固定数の事例を検索すればよい時には、類似度が高いSQL式から実行していき、出力されたレコード件数がその固定数に達したら、残りのSQL式実行を中断することも可能である。

5 おわりに

本手法において重要なことは実行速度である。利用者の1つの問い合わせを受け取ると、それを解釈して複数のSQL式に展開し、それらSQL式を実行する、という2段階の処理に分けることが出来るが、処理の大きさは、解釈・展開部分に比べてSQL実行部分が大きいので、本手法の実行速度は、商用データベースの実行速度に完全に依存する。実行速度については、我々も現在開発中の実用レベルのスケールの事例ベース検索システムで評価中であるが[5]、一般に商用データベースシステムでは、ワークステーション上で1秒間に2桁個程度のSELECT文実行が可能なので、実用上問題ないと考えている。

参考文献

[1] 島津ほか, "事例ベース推論システム構築用ツールの開発", 人工知能学会, 研究会,SIG-KBS-9102-7,1991年9月.

[2] 柴田ほか, "事例ベース検索システム構築用ツールの開発", 情処43全大,1991.

[3] 北野ほか, *"SQUAD：事例検索システム"*, 情処44全大,1992.

[4] H.Kitano,A.Shibata,H.Shimazu,J.Kajihara,A.Sato:"Building Large-Scale and Corporate-Wide Case-Based Systems:Integration of Organizational and Machine Excutable Algorithms",AAAI-92.

[5] 柴田ほか, "関係データベースを使った事例ベース検索 *(2)* - 実用システム", 情処45全大,1992