動的質問合成と文字列間の関連に基づくデータベース操作

3W－3

柴田典男　吉川正俊　植村俊亮

奈良先端科学技術大学院大学 情報科学研究科

1 まえがき

データベースに対して疑似自然言語から動的に対応するSQL文を合成するインターフェースの設計開発を行っている[1]。このような利用者インターフェースでは、文字列間の関連を定義することができ、質問改良やハイパーリンクなどの目的に利用できる。

本論文では、質問間の関連を抽出するための基本的な技法について述べる。また、文字列間の関連性を表現するため、2つの文字列間に**類似度**や**距離**の概念を導入する。また、システムが利用者の意図を反映した処理を行うための**視点**についても考察する。

2 質問の動的合成

ここでは利用者が与えた名詞句/節の疑似自然言語文に対応するSQL質問文を自動的に合成する機構の概略を与える。質問の動的合成を行なうための部品辞書には、データベーススキーマのクラスやメンバ変数などの構成要素を表す単語とそれに対応する質問部品を格納している。辞書は名詞節のためのもの(DictE)とそれらを連結する語句のためのもの(DictR)の二種類から成る。各辞書には文字列と対応するSQL文が`select`, `from`, `where`のそれぞれの句に分けて部品化して格納されている。入力文字列中の名詞句に相当する部分文字列間の包含関係をグラフで表現すると枝に順序のついた二分木となる。文字列からSQLへの変換はこの二分木を後順(postorder)で走査することにより順に行なう。

3 文字列の類似度

本章では、[1]で述べた文字列間の関連性をさらに厳密に考察し2つの文字列間の**類似度**を考える。この際、自然言語文を解析した木構造を保存しているならSQL文に変換しての比較が自然言語文の比較を包含してると考えられるため、本文ではそのような立場をとる。

3.1 視点の概念の導入

利用者の興味のある分野をシステムが理解していると便利である。本研究ではこの分野のことを**視点**と呼ぶことにする。

視点とは、複数個の単語の集合であり、各単語はスキーマ上のクラスやインスタンス、ビューとして存在しているものとし、利用者が興味を持っている分野をおおまかに表現する役割を持つ。システムがある利用者の視点を知るためには、利用者が興味のある分野を入力（またはメニューから選択）しなければならないが、利用者がいくつか質問を行なった後であれば、その質問の傾向から推測できる場合もある。

視点の利用方法としては、(1)質問改良する場合に、視点に遠い部分から変更や削除を行なう、(2)利用者の意図により近い文字列にハイパーリンクを張る、が考えられる。次節では、この視点を考慮に入れた類似度の定義について考察する。

3.2 視点を利用した類似度の定義

本節では、比較対象の名詞と視点との関連の深さによって類似度を定義できることについて述べる。

類似度を定義する際に、すでに合成されているSQL文を利用することができる。SQL文は`select`, `from`, `where`句にわかれており、それぞれの部分ごとに類似度が求まる。

SQL文の`select`句であらわされるのは、利用者が最も知りたい情報である。`select`句を比較して定義できる類似度は、(1)`select`句が同じ、(2)`select`句が共に視点の直系親族である、(3)`select`句の一方が視点の直系親族であり他方は視点の直系親族でない、(4)`select`句が共に視点の直系親族でない、であり、この順に類似度が小さくなっていく。直系親族とはクラスA,Bについて、AがBの先祖クラスまたは子孫クラスであることを指す。

`from`句を比較することで使用クラスの類似度の大きさをあらわすことができる。その際、それぞれの質問の`from`句に属するクラスの直積の数だけ比較を行なう。個々のクラスどうしを比較して定義できる類似度は、(1)同じクラス 、(2)共に視点の直系親族である、(3)一方が視点の直系親族であり、他方は視点の直系親族でない、(4)共に視点の直系親族でない、である。ここで(1)に重み1、(2)に重み0.75、(3)に重み0.5、(4)に重み0を与えて重みの合計を直積の数で割ることで類似度を求めるものとする。

`where`句を比較することで2つの文字列間に矛盾が生じる条件があるかどうかがわかる。その際、それぞれの質問の`where`句に属するクラスの直積の数だけ比較を行なう。個々の条件どうしを比較して定義できる分類は、(1)同じ条件 または 含意関係がある、(2)矛盾が生じる、(3)関係がない、である。他の分類に属する条件との比較は行わない。含意関係とは、条件P, Qに対し

Database Manipulation based on Dynamic Query Synthesis and Relationship between Character Strings
Norio SHIBATA, Masatoshi YOSHIKAWA and Shunsuke UEMURA
Graduate School of Information Science, NAra Institute of Science and Technology (NAIST)

てPがQに含意されるまたはQがPに含意されることを指す。さらにそれぞれの分類の中でも、(a) 両者とも視点の条件と含意関係がある、(b) 一方が視点の条件と含意関係があり、他方は含意関係がない、(c) 両者とも視点の条件と含意関係がない、の3通りの関係がありこの順で類似度が小さくなる。結局9通りの類似度が考えられる。

以上の結果をもとにSQLの類似度を求める。この際に、自然言語の単語単位で注目する場合は注目すべき単語のSQL部品だけを取り出すことで比較可能となる。また、SQLの構造に注目して比較する場合はどの句を優先するかを利用者が決定し、その句で類似度が判定できれば終り、さもなければ他の句をも考慮する。この優先順位は何も指定しなければ select 句、from 句、where 句の順とする。

例をあげると、視点が「仏教」で、「京都の寺院」と「奈良の仏像」を比較する場合、select 句は (2) の関連を持ち、from 句は 0.625 となり、where 句では (1)(c) と (3)(a)、という類似度を持つ。

3.3 視点を用いない類似度の定義

利用者が常に視点を入力するとは限らないし、また視点を使わずに類似度を知りたい場合もある。そこで、視点を用いない類似度の定義についても考察する。

select 句を比較することで、(1) select 句が同じ、(2) select 句どうしに is_a 関連がある、(3) select 句どうしに関連がない、の3種類が考えられるが、この順に類似度が小さくなっていく。

また、from 句を比較することで、クラス間の類似度の大きさを表すことができる。その際、それぞれの質問の from 句の直積の数だけ比較を行う。これらのクラスの間の関係は、(1) 同じクラス、(2) is_a 関係がある、(3) 関係なし、が考えられる。ここで「同じクラス」に重み 1、「is_a 関係がある」に重み 0.5、「関係なし」に重み 0 を与え、重みの合計を直積の数で割ることで類似度を判断できる。

さらに、where 句を比較することで2つの文字列間に矛盾が生じる条件があるかどうかがわかる。その際、それぞれの質問の where 句に属するクラスの直積の数だけ比較を行なう。個々の条件どうしを比較して定義できる類似度は、(1) 同じ条件 または 一方が他方に含意される (2) 矛盾が生じる、(3) 関係がない、である。ここで、(1) に分類された数から (2) に分類された数を引いたものを考える。この値が大きいほど類似度は大きくなる。

以上の結果をもとにSQLの類似度を求める。この結果の利用方法は前節と同じである。

例をあげると、「京都」と「奈良」の類似度は select 句、from 句は同じであるため、これらを基準にした場合には類似度は大きいが、where 句に矛盾が生じるため、where 句を基準にすると類似度は小さくなる。

3.4 文字列間の距離の定義

前節までで類似度について考察したが、任意の文字列間の関連を数値で表すことも考えられる。そこで、視点を考慮した文字列間の距離について考察する。なお、本節ではキークラス (利用者の最も必要とするクラス) に基づく距離の定義を考える。

(1) 名詞間の距離算出規則

2つの名詞間の距離を算出するには、スキーマ上を一方の名詞から他方の名詞へとトラバースする際に通過する枝の重みの和を求めればよい。ビューの場合はその辞書で登録されているクラスと同じとみなす。トラバースする際に、is_a の枝、属性の枝、クラスとインスタンスの間の枝が存在するが、これらの重みは利用者がどの部分を重要視するかによって可変である。キークラスを基準にした場合はそれぞれ、1, 3, 0.5 となる。また、ビューは辞書でクラスが表されているのでそのクラスと同等とみなしてよい。

(2) 名詞句間の距離の定義

名詞句間の距離を考える場合、前章で解析した木構造を考える。一般には2つの文字列の木構造が異なるため、再帰的に左の部分木どうし、右の部分木どうしを比較していくと最終的に「葉」と「葉」、または「葉」と「(部分) 木」の距離を求めることになる。そこで、得られた距離をすべて合計すればよい。「葉」と「葉」の距離は上での算出規則に従う。「葉」と「木」の距離を算出するには、まず「葉」と「木の一番右の葉」との距離を算出し、「一番右の葉」以外の「葉」の数だけ距離を増やすという規則を適用する。

(3) 部分文字列を含む場合の処理

これまでの議論では現実的ではない距離が定義される可能性がある。そこで、部分文字列を含んだ場合については距離を測定する際に、質問全体のキークラスに関わる場合は部分文字列を含んでいても考慮に入れず、「葉」と「葉」の場合には距離を0とし、「葉」と「木」の場合にはその「葉」と「木」との距離は1とする、というルールに従って距離を測定するように修正する。

4 あとがき

今後はこのような機能を持つデータベースインターフェースの設計を進め、実装を行なう。また、実装する際に生じる問題点を基により使いやすいシステム設計を行なう。

謝辞

日頃から数々の有益な御討論をいただく植村研究室の皆様に感謝致します。

参考文献

[1] 吉川正俊, 柴田典男, 植村俊亮:動的質問合成に基づくオブジェクトベース利用者インターフェース, 情報処理学会第 48 回全国大会 1E-5,1994.