5G-5 エキスパートシステム構築ツールＩＲＥＸにおけるデータベース結合機能

荒木　大*1　　小島昌一*1　　大森和則*2
(株)東芝　*1システム・ソフトウェア技術研究所　*2府中工場

1．まえがき

大規模なエキスパートシステム（ＥＳ）を構築する際には、知識の記述量も膨大なものとなることが予想される。既存のデータベースにすでに蓄積されているデータを、ＥＳの実行時の知識として利用することは、この問題を解決する一つの有効な手段と言えよう。

本稿では、ＥＳ構築ツールＩＲＥＸ[1]とリレーショナル・データベース（ＲＤＢ）との結合を考え、ＩＲＥＸの知識表現上でのＲＤＢの表現形式と、両者のインターフェースの実現方式について述べる。

2．ＩＲＥＸとＲＤＢの結合へのアプローチ

ここでは、既存のデータベースを、知識ベースとしてＥＳから利用することを可能にするために、外部のデータベース管理システムを使用して、知識ベースと外部のＲＤＢとの統合化を行う機能をＩＲＥＸに設ける。

ＩＲＥＸでは『if...then...』形式のプロダクションルールを知識表現の一形式として用いており、プロダクションルールと事実情報を格納しているワーキング・メモリ（ＷＭ）によって推論が進められる。従って、ＩＲＥＸとデータベース管理システムを接続し、実体としては外部のＲＤＢに格納されているデータを、知識表現上ではＷＭの形で扱えるようにすることによって、ＲＤＢ中のデータも推論実行時の事実情報の一部として利用できるようになる。ここではＲＤＢの関係表のタプルごとに、同一のクラス名を持つＷＭで表現するものとし、ＲＤＢ中の関係の定義にはＳＱＬ言語を使用する。

ＲＤＢとの結合を実現する方法の第一のアプローチとして、ＲＤＢをアクセスするコマンドをＩＲＥＸに設ける方法がある。ユーザがプロダクションルールの操作部でこのコマンドを記述しておくと、そのルールが発火した時点でＲＤＢを検索して結果がＷＭに展開される。しかし、この方法ではＲＤＢ検索を起動するタイミング及び作成したＷＭの管理はユーザに任されることになり、次の問題点を生じる。

(1) 推論実行中に何度もＲＤＢに対して検索をする場合、ユーザが十分な管理をしていないと同じＷＭが作られる。

(2) 検索結果が大量にあった時には、ＷＭも大量に作られて推論実行効率を損なう。また、以後の推論には不必要なＷＭ（１度もパターンマッチに成功しない）もここで作られる可能性がある。

そこで、ＲＤＢのデータを仮想ＷＭ化する機構、すなわち、ＷＭとＲＤＢとの対応関係をユーザが定義するだけで、ＲＤＢ中のデータがＷＭの形の事実情報としてプロダクションルールの条件部で使えるようにするアプローチが望ましい。このためには、ＲＤＢを検索するタイミング、およびＲＤＢを検索した結果の管理を制御する必要がある。

以下では、この第二のアプローチによるＲＤＢとの結合を考え、知識表現上でのＲＤＢの表現形式と、検索処理の効率的な実現方法を考える。

3．ＲＤＢの表現形式

知識表現上では、ＲＤＢ中に存在するデータは特別なＷＭ要素（ＲＤＢＷＭ）で表現する。このＷＭ要素はユーザがルールの操作部で作成、修正、削除の対象にすることができないＷＭ要素とする。ＲＤＢＷＭのデータ構造と、ＳＱＬ言語で記述したＲＤＢの関係との対応関係は、ＷＭ要素の宣言部で次の例の形式で記述する。

```
RDB(名簿          コード:int
                  名前  :char
                  年齢  :int      )
{
        select  code, name, age
        from    register
        where   age >= 30
}
```

Fig.1. ＲＤＢＷＭ要素の宣言例

上の記述中には、クラス“名簿”のＷＭの属性“コード”の値と、ＲＤＢ中に存在するテーブル“register”のフィールド“code”の値が対応するといった対応関係が定義されている。

また、where句以下は省略できるが、例のように推論に用いるＲＤＢのスコープを予め限定しておくこともできる。

Interfacing Databases With IREX
Dai ARAKI, Shoichi KOJIMA, Kazunori OHMORI
TOSHIBA Corp. *1Systems & Software Engineering Lab. *2Fuchu Works

上記の形式でＲＤＢＷＭを定義することによって、次の形の条件部を持つルールが書ける。

```
(A ^a1 〈x〉 ^a2 〈y〉)  /* 従来のＷＭ要素 */
(名簿 ^コード 〈= 〈x〉 ^名前 〈z〉 ^年齢 〈y〉)
   --〉 ...
```

Fig.2. ＲＤＢＷＭを用いたルール

このルールは、属性a1に1000、a2に50を値として持つクラスＡのＷＭ要素がある時は、コードが1000以下で年齢が50のデータをＲＤＢ中で検索し、該当するデータがあれば発火する。

4．ＲＤＢ検索のタイミングおよび検索結果の扱い方

次に、3.で述べた機能の実現方法を考えてゆく。ここでは、オーバーヘッドの大きいＲＤＢ検索の回数をできるだけ減らすことと、ＷＭに変換して蓄積しておくデータは必要最小限にすることを目標とする。

ＩＲＥＸではプロダクションルールの条件部をReteネットに展開してＷＭのパタンマッチングを行う。ＲＤＢ検索のタイミングはandノードで制御することを考える。

例として２つのルールを考える

```
rule1 {
        (A ^a1 〈x〉 ^a2 5)
        (名簿 ^年齢 =〈 〈x〉)
        --〉 ...
}
rule2 {
        (A ^a1 〈x〉 ^a2 3)
        (名簿 ^年齢 〈x〉)
        --〉 ...
}
```

Fig.3. ルール記述例

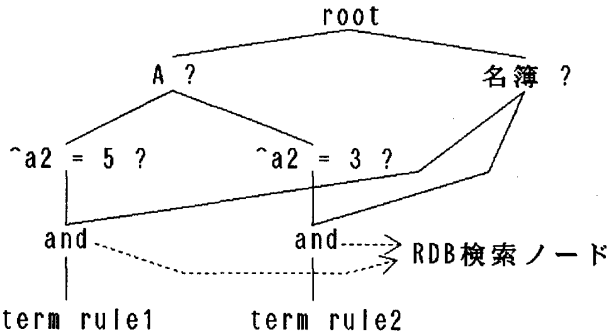

Fig.4. ＲＤＢ検索ノードを持つReteネット

Fig.3.のルールに対するReteネットをFig.4.に示した。ＲＤＢ検索ノードはＲＤＢＷＭのクラス単位で作成し、このノードは、ＲＤＢＷＭの要素間特性を評価するandノードから起動される。

「andノードの動作」

通常のandノードとしての処理に加えて、図で左からトークンが到着した時には、上位での変数の束縛状況に応じた条件をＲＤＢ検索ノードに渡す。

「ＲＤＢ検索ノードの動作」

andノードから条件を受けて起動すると、次の検索条件でＲＤＢにアクセスする。

（宣言部で定義されている条件）
and（渡された検索条件）
and（前に渡された検索条件の否定の積）

ただし、この条件が空の場合は新たな検索は行わない。返されてきた検索結果はＷＭに変換し、そのトークンをReteネットのrootから流す。

andノードはReteネット内では比較的下位に位置する。このため、ＲＤＢＷＭに関する評価、すなわち、ＲＤＢ検索を必要になるまで遅らせる効果が得られる。

5．評価

4.で述べた実現法は、初期的なパタンマッチはＲＤＢ上で行い、一度パタンマッチしたデータは再び使われる可能性が多いので、ＷＭとしてＥＳ内に残すことを方針としている。

これに対して、ＲＤＢＷＭのパタンマッチが必要になった時点でＲＤＢを検索するが、結果をＥＳ内に残さないでおく方法がある。この方法はメモリ効率の点では優れるが、例えば4.で挙げた例でクラスＡのＷＭ要素が多数存在するといった状況では、全く同じ条件で何度もＲＤＢを検索する可能性が高く、無駄な処理が多いと言える。

検索結果をＥＳ内に残す場合には、前の検索で既に取り込んだデータと重複するデータを再度取り込むことのないように、過去の検索条件を残しておく必要がある。しかし、結果として全体的な検索回数が減るため、推論実行効率は高まる。

6．まとめ

ＥＳ構築ツールＩＲＥＸにおいて、ＲＤＢ中のデータを仮想ＷＭ化する機能を述べた。

この方法では、ＲＤＢを検索するタイミングおよび取り込んだデータの管理は完全にＩＲＥＸが行うために、ユーザがこれを陽に制御する必要が無く、ＲＤＢへのアクセス回数も低く抑える効果が得られる。

今後の課題としては、ＩＲＥＸの推論結果をＲＤＢに動的に反映させてゆく機構の考察がある。

7．参考文献

〔1〕河野ほか，『エキスパートシステム構築支援ツールＩＲＥＸ(1)～(5)』，情報処理学会第34回全国大会予稿集，1L-1～5