帰納論理プログラミングにおける関係データベース技術を用いたアルゴリズムの提案

池田 晃* 難波 和明† 大和田 勇人†

東京理科大学大学院 理工学研究科 経営工学専攻*
東京理科大学 理工学部 経営工学科†

1 はじめに

近年，ストレージの大容量化やウェブ技術の発達により，膨大な情報をまとめて収集・蓄積できるようになってきた．その一方で，大量のデータを有効に活用することは難しく，膨大な収集データから特定の知識を抽出できるデータマイニングに注目が集まっている．

その中でも，帰納論理プログラミング（Inductive Logic Programming:ILP[1]）は，一階述語論理を表現言語としており，他の手法に比べてより細かな推論規則を導ける．命題論理では解を導けない問題に対しても，マルチリレーショナルデータマイニングを適用することが可能であるため，高度な学習が必要なバイオ分野等への活用が期待されている．

ただILPは他手法に比べて精度が高いものの，探索対象となる仮説空間が膨大になるため，メモリに関する制約から大規模な問題を扱うことができない．元のデータに対して，メモリに展開されるデータ量は大幅に増えてしまうのである．

本論文の目的は，背景知識をメモリ内に全て格納することなく，ILPを動作させる仕組みを提案することで実問題への適用を容易にすることである．提案手法では背景知識の格納及び被覆演算の大半を関係データベースが行うため，従来型のILPシステムを一切用いないで，一から実装している．尚，背景知識を関係データベース上で扱う都合上，再帰的な表現を含む問題などは一部適用対象外となるが，基本的には従来型ILPシステムと同等の学習能力を有している．

2 提案手法

2.1 アルゴリズム

本論文では，探索におけるメモリ容量の制約を問題視しているため，あらかじめ関係データベース上に各事例及び背景知識を格納することにより，ローカルのメモリ空間を占有することなく，探索と評価を行っている．

関係データベースには事例および背景知識を述語別に表として保持しておき，背景知識の属性など最低限の関連情報だけをローカルに保存しておく．実行する際にはその都度データベースからデータを取得し，ローカルで情報の処理をして，再びデータベースへと書き戻している．

以下に適用するアルゴリズムの概略を示す．**STEP1**～**STEP3**まで（一段階目）と**STEP4**以降（二段階目）で大きく分かれている二段階アプローチとなっている．

STEP1　データベースから全事例について，関連する背景知識を全て取得し，それぞれ最弱仮説を生成する
STEP2　正事例，負事例ごとに分け，それぞれ全事例に適合する背景知識を集計する
STEP3　正事例側に含まれる知識（述語）から，負事例側にも含まれるものを取り除き，候補仮説とする
STEP4　候補仮説と各事例を比較し，負事例を含まなければ終了し，含む場合は任意の背景知識を付加し，新たな仮説の候補とする
STEP5　新たな候補仮説が条件（全正事例を被覆する）を満たせば終了し，満たさない場合は前回の仮説候補に戻して**STEP4**を実行する

一般的にILPの探索アルゴリズムでは，まずある仮説を生成し，その仮説を元にして選択と精密化を繰り返していく．提案手法では一段階目において元となる候補仮説を導いている．二段階目の選択及び精密化に時間を要するため，真の解に少しでも近い候補仮説を導くことが重要である．そこで一段階目のアルゴリズムに関係データベースの特性をうまく組み込むことによって，精度の高い仮説を生成し，最終的に二段階目で真の解を導くあるいは証明するという流れになっている．

2.2 システム

本システムはローカルから指示を出してデータベースを操作し，取得したデータに対してローカルで必要な処理を行った上で，再びデータベースに書き戻す，といった一連の作業を行っている．したがってデータベースとの連携がしやすいよう，JavaとMySQL（RDBMS）の連携をメインにシステムの構築を行っている．これにより理論的に，処理を担うクライアント側から，外部のデータベースサーバにアクセスしてデータを処理することが可能である．

全ての表は正規化されているので，集計処理が容易である．特にアルゴリズムの**STEP2**においては，まず初期仮説を取

Suggestion of the algorithm using relational database technology in inductive logic programming
Hikaru IKEDA*, Kazuaki NAMBA†, and Hayato OHWADA†
Department of Industrial Administration, Graduate school of Science and Technology, Tokyo University of Science*,
Department of Industrial Administration, Faculty of Science and Technology, Tokyo University of Science†

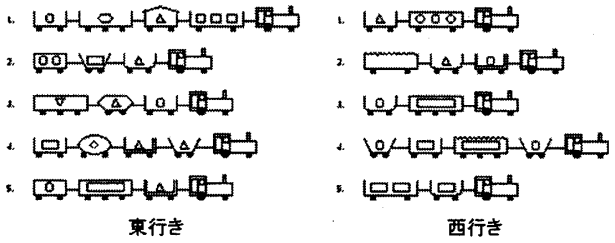

図1　Michalskiの列車問題

表1　実行時間（秒）の比較

事例数	提案手法	従来手法
100	0.649	0.096
1000	8.547	0.869
10000	608.476	41.359
100000	59239.406	7353.670

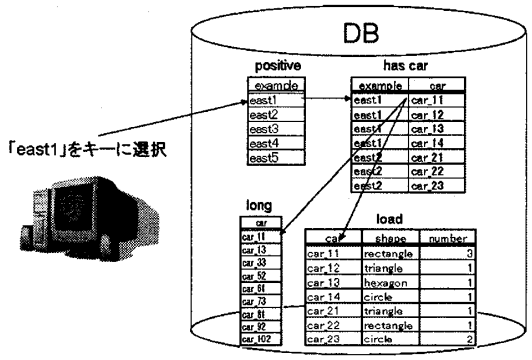

図2　列車問題におけるシステムのイメージ

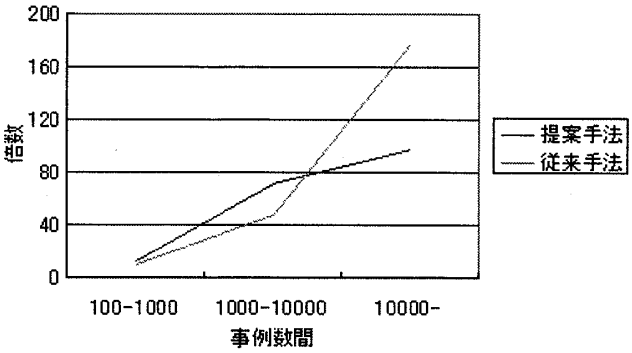

図3　事例数の増加と時間増加率の比較

得したクエリの結果をローカルで集計し，データベースへ書き戻して表を作成する，そこにSELECT COUNT(*)から始まるSQL文のクエリを処理するだけで，候補仮説の導出が終了する．このように仮説空間を圧縮する処理が一度の実行で完了するため，探索領域の効率的な削減が可能である．膨大な量の背景知識をデータベースに保持しておくため，あらかじめ全データをメモリにロードする時間を省くと共に，メモリの容量について気を配る必要もなくなる．

3　実験

3.1　実験データ

今回，実験データとしてILPにおけるMichalskiの列車問題[3]（図1）を用いた．データ（事例）数は100・1000・10000・100000の4通りで実験を行った．いずれのデータでも正事例・負事例が半分ずつで，得られるルールは全て同じものである．データは全てマルチリレーショナルデータとしてMySQLへ格納されており，ローカルからSQL文による問い合わせをすることで，データ処理を行っている．

3.2　実験手法

既存のILPであるProgol[2]と実装したシステム（図2）で同様の実験を行った．前者はProlog形式で作成されたデータをローカルに保持しておき，後者は3.1のようにMySQLに格納されたデータを随時読み出している．尚，実行時間の計測にはLinuxの`/usr/bin/time`における「real」を用いている．

4　結果

全ての実験において，正しくルールを導くことができた．また，提案手法では第一段階だけで正解を導くことができたため，提案手法の実行時間は第一段階終了時点のものを使用している．実行時間についての比較結果を表1と図3に示す．

図3より，提案手法はデータが大幅に増加しても，実行時間の増加率は低く抑えられていることがわかる．その反面，従来手法では実行時間の増加が指数関数的に伸びており，一般的なILPシステムの弱点が露呈される結果となっている．

5　結論

関係データベース技術を用いた提案手法は，従来手法と同等の推論能力を有することがわかった．また提案手法はデータの規模が大きくなるにつれて，相対的に高速化するため，大規模データに適したアルゴリズムであり，ILPにおけるメモリ上での問題を解決する有用な手法であると言える．

参考文献

[1] Inductive logic programming: theory and methods, S.H. Muggleton and L. De Raedt. Journal of Logic Programming, Vol.19/20,pp. 629–679, 1994.

[2] Inverse entailment and Progol, S.H. Muggleton. New Generation Computing, Vol.13,pp. 245–286, 1995.

[3] Michalski, R.S., Larson, J.B.: Selection of Most Representative Training Examples and Incremental Generation of VL918 Hypotheses. Technical Report 867, Department of Computer Science, University of Illinois, 1978.