帰納論理プログラミングにおける 仮説探索の効率化・大規模化の比較検討

Comparing with searching hypothesis efficiently and massively in inductive logic programming

池田 晃[*1] 松井 藤五郎[*2] 大和田 勇人[*2]
Hikaru Ikeda Tohgoroh Matsui Hayato Ohwada

[*1]東京理科大学大学院理工学研究科経営工学専攻
Department of Industrial Administration, Graduate school of Science and Technology, Tokyo University of Science

[*2]東京理科大学理工学部経営工学科
Department of Industrial Administration, Faculty of Science and Technology, Tokyo University of Science

We would compare among the methods to resolve the problems of inductive logic programming which need enormous hypothesis space and stack memories. We selected GOLEM based argorithm to reduce hypothesis space. Also, it is very useful to use the RDBMS in order to compute the processes efficiently. In our method, it must be possible to compute the enormous data.

1. はじめに

近年，コンピュータのストレージやウェブ技術の発達により，比較的容易に膨大な情報を収集することができるようになってきた．その一方で，大量のデータを整理して活用することは難しく，情報を蓄積するだけで有効利用できていない事例は少なくない．こういった膨大なデータから知識を抽出するデータマイニングは消費者の趣向が多岐にわたる現代のマーケティングで重要視されている．

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，他の手法に比べてより細かな推論規則を導けるため，現実的な問題を扱う際に強力な手法であると言われており，代表的なシステムとして Progol[2] や GKS[3] などがある．ILP は他の手法に比べて精度が高いなど優れた面が多いが，探索の対象となる仮説空間が膨大になるため，学習速度が遅かったり，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かったり，という問題がある．したがって限られた事例下では成果を上げることができるものの，実問題への適用にあたってはメモリに格納できる背景知識の量に制限され，大規模な問題に対して応用された例は少ない．

本研究の目的は，背景知識をメモリに格納することなく，ILP を動作させる仕組みを作ることで実問題への適用を容易にし，ILP の利用範囲を広げ，汎用性の高いシステムの構築につなげていくことである．大規模なデータにも適用可能なシステムを目指すため，従来型 ILP の限界である 100MB 以上のデータ適用を目標とする．尚，ILP で扱える問題の中でも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目指していく．

2. 探索手法

ILP における探索手法にはトップダウン型とボトムアップ型の大きく分けて二種類が存在する．両者の違いは必要とするメモリの大きさにも影響するため，簡単に説明しておく．

連絡先: 池田 晃，東京理科大学大学院理工学研究科経営工学専攻，千葉県野田市山崎 2641，E-mail:j7408602@ed.noda.tus.ac.jp

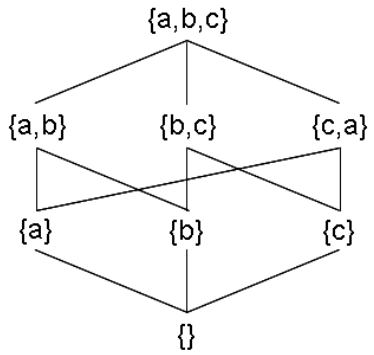

図 1: べき集合の束による表現

2.1 仮説空間

仮説空間は，すべての仮説を構造化した空間であり，ILP が最適な仮説を求めて探索を行う．仮説空間を，仮説間の一般化関係によって構造化すると，後の探索が楽になる．一般化関係は，束と呼ばれる数学的構造をしていることが多い．

束（lattice）とは簡単に言えば，任意の二つの要素が必ず共通の祖先と共通の子孫を持つ構造のことである．最も近い共通の子孫はその二つの要素の最大下界（greatest lower bound）あるいは下限（infimum）と呼ばれ，最も近い共通の祖先は最小上界（least upper bound）あるいは上限（supremum）と呼ばれる．最大下界を求める演算を結び（meet）と呼び，記号 $\sqcap$ で表す．また，最小上界を求める演算を交わり（join）と呼び，記号 $\sqcup$ で表す．

集合の包含関係によって束が形成されており，下界（$\{\}$）は空集合で，上界（$\{a, b, c\}$）に近づくに従って要素数が増えている．この束での結びは集合積 $\cap$ で，交わりは集合和 $\cup$ である．このことは，図 1 に示すように，下界を下に，上界を上に置いた形で表される．

ILP における仮説空間は，仮説空間の一般化関係によって，多くの場合，束となる．束を成すための条件は，任意の二つの要素の上限と下限が存在することである．それらの下限の存在は，容易に示されるため，問題となるのは，上限が存在するた

めの条件を明らかにすることである．

2.2 探索手法

ILP はすべての仮説空間の中から，一般性基準に基づく束を導入し，ある探索アルゴリズムに従って，より良い仮説を見つけていく．探索においては，仮説の選択と精密化を行う必要があり，精密化の側面では，束の上界から下界に向かって調べていくトップダウン型とその反対に下界から上界へと探索しているボトムアップ型がある．すなわち，受け取った仮説よりも特殊な仮説集合を生成する（トップダウン探索）のか，逆に，より一般的な仮説集合を生成する（ボトムアップ探索）のかの違いである．

トップダウン探索は，最も特殊性の高い最弱仮説を元にして，少しずつ一般化していく過程で解を探索していく手法であり，代表的なツールに Progol などがある．あらゆる仮説を網羅的に探索することになるので，大量のメモリを食ってしまうという欠点があり，大規模な問題への適用が困難である．

それに対して，ボトムアップ探索は，最も一般的な初期仮説から始めて，徐々に特殊化を進めていき，それ以上一般化できなくなった時点で，他の初期仮説を検討していくもので，GOLEM[4] などが有名である．問題の束構造によっては，膨大な時間を要することもあり，探索手法を検討する上で，十分考慮しておく必要がある．

3. 提案手法

提案手法について，適用するアルゴリズムとシステムの実装環境という両面から説明する．

3.1 適用アルゴリズム

本研究では，探索におけるメモリ容量の制約を最も問題視しているため，メモリを大量に消費する必要のない GOLEM のアルゴリズムを応用する．探索空間の増大を抑えることで，探索時間とメモリ容量の双方を削減し，効率的な探索を実現する．以下に適用するアルゴリズムの概略を示す．

STEP1 背景知識及びデータ構造を外部のデータベースに格納

STEP2 データベースから SQL 文により条件を満たす背景知識を取得し，初期仮説を生成

STEP3 初期仮説を元に条件を追加して仮説を生成し，合致する正事例の数が増えれば採用し，減れば採用しない

STEP4 STEP3 を繰り返し，正事例を包含する数が閾値以上であり，最も特殊な条件になれば終了する

初期仮説の選択の仕方によって，演算時間は大きく変わる可能性があるが，あらかじめ特定することは困難であるため，ランダムに生成する．条件の付加は一つずつ全組合せについて行い，最も適合度の高い仮説を選択する．被覆数の算定や適合する条件の事例選択は RDBMS が担い，join 演算や count 関数の活用によって演算効率の改善を狙う．あらかじめ問題の特性に基づいて設定しておいた閾値の範囲内で，最も特殊な条件となったときの仮説が答えとなる．

3.2 システムの概要

本手法では，計算機のメモリ容量に制限されることなく，システムを動作させるため，背景知識をすべてメモリに格納することはせず，外部の関係データベースに保持しておく．すなわち，関係データベースからの直接学習である．システムのイメージを図 2 に示す．

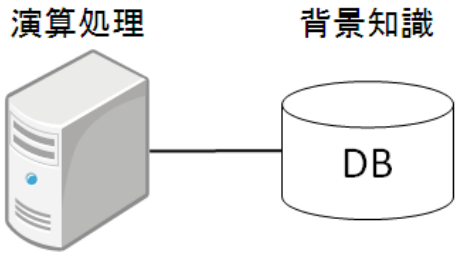

図 2: システムのイメージ

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，あらかじめデータをメモリにロードする時間を省き，もちろん，メモリの容量について気を配る必要もなくなる．極論を言えば，ハードディスクの容量限界まで背景知識を格納できることになる．これによって，背景知識の容量による制約が大幅に緩和される．

ただしデータベースを用いる場合，仮説を導く際に必要な背景知識をその都度，メモリに読み込んでこなくてはならない．メモリへとロードしてくる時間は余分にかかってしまうものの，全背景知識から該当のデータを検索するコストを抑え，データベース管理システム（DBMS）に検索を任せることができる．検索処理に適した構造となっている DBMS を利用できるため，検索効率の改善が見込まれ，トータルとしての大幅な時間短縮が期待される．

4. まとめ

本論文では，大規模な問題に対して，ILP を適用するために必要なアルゴリズムについて提案した．現時点ではシステムの実装には至っておらず，本手法の有効性は未知数であり，検証が必要である．今後は，本手法の有効性を確かめるために，システムを実装した上で，他手法との比較実験を行い，ILP の実問題への適用を実現させていきたい．

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] Concurrent Execution of Optimal Hypothesis Search for Inverse Entailment, Ohwada. H.,Nishiyama. H.,Mizoguchi. F., Lecture notes in Artificial Intelligence, Springer-Verlag, No.1866, pp.165-173, 2000.

[4] Efficient induction of logic programs, S.H. Muggleton, C. Feng. Inductive logic programming, Academic Press, 1992.