人　工　知　能　103－8
（1996．1．17）

グラフによって記述された事例集合からの MDL 基準による階層構造の発見

福田慶郎　　上原 邦昭

神戸大学 工学部 情報知能工学科

内容梗概

CBR システムにおける事例の記述言語は，単純な属性値ベクトル表現から複雑な構造的表現までの多岐に渡っている．特に複雑な問題領域においては，事例の記述言語に高い記述力と拡張性が要求されている．グラフ表現は記述力と拡張性に優れた記述言語の一つであるが，照合などにかかる計算量が属性値表現に比べて大きいという問題点がある．本研究では，MDL 基準に従って最も効率良くグラフ集合を記述する部分グラフの階層構造を構築し，事例の照合にかかるコストの軽減をはかる手法を提案する．さらに，二つのグラフ間のゆがみの記述長を定義し，柔軟なグラフマッチングを用いて階層構造を縮小する手法を提案する．

Hierarchical Organization of Graph Structured Cases Using MDL Principle

Yoshio Fukuda　　Kuniaki Uehara

Department of Computer and Systems Engineering, Kobe University

Abstract

CBR systems have their own case representations, such as set of attribute-value pairs, feature vector, list of propositions, or graph. Especially, in a complex domain, a more expressive and flexible case representation is desirable. A graph structured representation is one of the most expressive and flexible representations. However, the retrieval process of cases represented by graph is computationally intractable. In this paper, in order to reduce the matching cost, We will propose the method to organize a set of graph structured cases into a hierarchy structure using the minimum description length principle. Furthermore, we will also show that the size of the hierarchy structure can be reduced using an inexact graph matching technique.

1 はじめに

事例ベース推論 (Case Based Reasoning, 以下 CBR と略す) は，Schank らの提案した人間の記憶のモデルに関する研究に基づいている．CBR システムは，過去の経験を事例として蓄えており，新しい問題に対して過去の類似した事例を検索し，その事例を利用して新しい問題に対処するものである．

CBR システムの多くは，属性と値のペアからなる属性値表現を用いて事例を記述している．属性値表現は，特徴間の関係を記述できないために表現力が低く，複雑な対象を十分に記述することができないという問題がある．この問題に対して，述語論理やグラフ表現などの柔軟で構造的な表現で事例を記述し，事例の表現力と拡張性をともに向上させる研究がなされている．

グラフ表現は表現力に富む反面，照合にかかる計算量が属性値表現や限定された構造を持つ表現に比べて大きいために，事例検索の効率改善に関する研究がなされている [1][2]．一つのアプローチはアルゴリズムの並列化であり，もう一つは事例集合の組織化である．本研究は後者のアプローチをとっている．すなわち，複数の事例に共通する特徴に従って事例集合を組織化し，複数の事例の共通部分に対する照合を一度で済ませ，事例検索の効率を改善するというアプローチである．一般に，事例集合の組織化問題では，事例集合があらかじめ与えられる場合には構造の最適化が問題となり，事例が逐次的に与えられる場合には構造のインクリメンタルな更新アルゴリズムが必要となる．本研究では，事例集合はあらかじめ与えられると仮定しており，グラフ表現された事例集合を最適な階層構造へ組織化することを目的としている．

2 事例集合の階層構造化

2.1 事例集合の組織化と検索

グラフ表現された事例集合を，図 1 の実線で示しているようなネットワークへ組織化し，検索効率を改善する研究が Bunke によってなされている [2]．その中で，事例集合の組織化に関する指針がいくつか挙げられている．

1. 事例全体に共通する最大の構造を階層構造の中に組み入れる

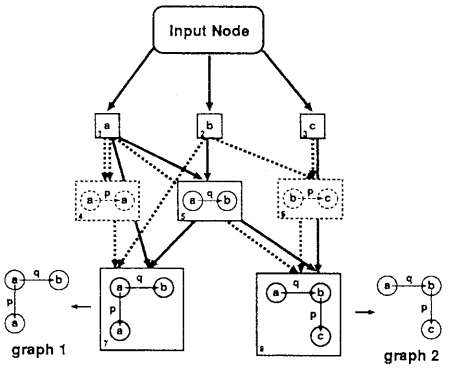

図 1: ネットワークの例

2. 二つの事例のペアに共通する最大の構造を階層構造の中に組み入れる

3. 優先度の高い事例の構造を階層構造の中に組み入れる

4. 全体としてバランスのとれた階層構造を選択する

事例集合の組織化では，事例があらかじめ与えられる場合は最適構造の選択が問題となり，逐次的に与えられる場合は動的に記憶構造を変更するアルゴリズムが必要となる．Bunke の NA は逐次的なアルゴリズムであり，3 と 4 の方針をとっている．具体的には，新しい事例を事例集合に加える際に，事例集合を表現するネットワークに事例を入力し，ネットワークで表現できない部分を複数の新しいノードとして逐次的にネットワークに加えるようにしている．図 1 を用いて説明する．事例として graph 1 だけを記憶していると仮定すると，ネットワーク中に含まれるノードは 1, 2, 7 と 4 あるいは 5 だけである．仮にノード 5 が生成されているとして，ネットワークに新しい事例である graph 2 が入力されると，現在のネットワークで表現できない部分として，ノード 3 と 8 を生成しネットワークに加える．なお，現在のネットワークで事例を表現できる場合には，ネットワークの変更は行なわれない．また，新しいノードはネットワークのバランスを保つように (二つの親ノードのサイズが同サイズになるように) 加えられるようにしている．

一方，本研究では，1, 2 の方針に重点を置いている．すなわち，事例はあらかじめ与えられると仮定して，事例集合の組織化を一括処理で行ない，最適な構造を探

索するようにしている．方針 1 で採用される共通部分は全事例に共通しており，全ての事例が同じ構造を持っている．また，方針 2 で採用される部分構造を持つ事例は，方針 1 に比べて構造が大きくなる．共通部分を利用して事例集合を組織化する目的は，事例集合の複雑さを解消し，検索効率を改善することにある．したがって，方針 1 の構造を用いると，同じ構造を持つ事例は複数存在するが，構造自体が小さいために複雑さはそれほど解消されない．逆に，方針 2 の構造を用いると，構造自体が大きくなり，その構造を持つ事例は単純化されるが，その他の事例の複雑さは解消されない．

このような問題点を解決するために，本研究では MDL (Minimum Description Length) 基準に従い，事例集合を最も単純に記述する部分構造の組合せを探索するアプローチをとっている．MDL 基準の採用により，方針 1 と 2 のトレードオフをはかり，事例集合を最も効率良く表現することのできる大きさと数を持つ部分構造を選択することができるようになるという利点が生じる．

2.2 MDL 原理による階層構造の評価

MDL 原理は，データから未知の情報源の確率モデルを推定するための選択原理の一つとして，情報理論の枠組で提案されたものである [4]．MDL 基準における最良のモデルとは，データを説明するモデル自身の記述長とそのモデルを用いてデータを記述した際の記述長の和が最小となるようなモデルである．

本研究では，MDL 基準により，部分グラフの組合せである階層構造を用いてグラフ集合を表現した際の記述長でその構造を評価し，最も記述長を短くする部分グラフの組合せを最適な構造として選択する手法を検討する．

まず初めに，本研究で用いているグラフの符号化方法を説明する．あるグラフ G の節点の数を v，節点の持つラベルの数を l_v，辺の数を e，辺の持つラベルの数を l_e とすると，グラフの記述長 $DL(G)$ は

$$\begin{aligned} DL(G) &= log_2 v + v log_2 l_v + log_2 e \\ &\quad + e(log_2 l_e + 2log_2 v)[bits] \end{aligned} \quad (1)$$

となる．部分グラフ SG を用いて元のグラフ G を記述した時の記述長 $DL(G|SG)$ は以下のように計算できる．SG の数を sg，節点の数を v_{sg}，辺の数を e_{sg} とすると，

$$\begin{aligned} & DL(G|SG) \\ &= log_2(v - v_{sg} + sg) \\ &\quad + (v - v_{sg} + sg) log_2(l_v + 1) \\ &\quad + log_2(e - e_{sg}) \\ &\quad + (e - e_{sg}) log_2 l_e \\ &\quad + (e - e_{sg}) 2log_2(v - v_{sg} + sg)[bits] \end{aligned} \quad (2)$$

となる．$DL(G_1|SG) + DL(G_2|SG) + \cdots + DL(G_n|SG) + DL(SG)$ を最小にする SG は，グラフ集合 $G_1, G_2, \cdots, G_n$ を最も効率良く表現する部分グラフである．

本研究では，このような符号化手法を用いて，部分グラフの組合せによるグラフ集合の記述長を評価している．さらに，最も記述長を短くするような部分グラフの組合せを，グラフ集合を効率良く表現する部分グラフの階層構造として選択するようにしている．

2.3 部分グラフの組合せの探索

本研究と同じく，MDL 基準を用いてグラフ集合から部分グラフを発見するシステムに SUBDUE がある [5]．SUBDUE は，ビームサーチによりランダムに選択した初期節点を初期モデルとし，最も記述長の短くなる節点を選んで，モデルを拡張するシステムである．記述長が改善されなくなるまで拡張を繰り返し，最も記述長の短いモデルが一つだけ部分グラフとして採用される．発見した部分グラフを用いて記述したグラフ集合に対して，同じプロセスを繰り返し，部分グラフによる階層構造が構築される．しかし，SUBDUE で探索される階層構造は最適ではない．

このような問題に対して，本システムはグラフ集合を最も効率良く表現する部分グラフの組合せである階層構造を発見することを目的としている．具体的には，まず図 1 の点線で示した部分を含むような，全ての部分グラフの可能性を含むネットワーク (図 1 で示したネットワーク) を作成し，次に MDL 基準によりグラフ集合を最も効率良く記述するノードの組合せ (図 1 の実線で示した部分) を探索する．そして，探索された組合せに含まれないノード (図 1 の点線で示した部分) をネットワークから削除する．以上の動作のうち，全ての部分グラフの可能性を含むネットワークを作成するアルゴリズムを図 2 に示す．図 2 中のアルゴリズム

```
Algorithm Compile(GraphSet);
begin
  Subgraphs := MakeVertexNodes(GraphSet);
  Combinations := MakeCombination(Subgraphs);
  while Combinations ≠ NULL
    G_i, G_j := pop(Combinations);
    MakeSubgraphNodes(G_i, G_j);
  endwhile
end.
```

```
Algorithm MakeSubgraphNodes(G_i, G_j);
begin
  I_i := Instance(G_i);
  I_j := Instance(G_j);
  Instances := Combination(I_i, I_j);
  while Instances ≠ NULL
    Instance := pop(Instances);
    Edges := CheckEdges(Instance);
    if Edges ≠ NULL then
      I := Search(Instance, Instances);
      Instances := Remove(I, Instances);
      G := Include(Instance, Subgraphs);
      if G == NULL then
        G := NewNode(G_i, G_j, I);
      endif
      G := ChildNode(G_i, G_j);
      Combinations := Update(G, Combinations);
      Subgraphs := append(G, Subgraphs);
    endif
  endwhile
```

図 2: ネットワーク作成アルゴリズム

Compile により，グラフ集合から図 1 で示しているようなネットワークを作成する．

このアルゴリズムにより作成されたネットワークは，全ての部分グラフの可能性を含むために，ノード数が多く冗長なものとなっている．したがって，グラフ集合を最も短く記述する部分グラフの組合せを探索し，それ以外の不要なノードををネットワークから削除する必要がある．この部分グラフの組合せ探索アルゴリズムを図 3 に示す．

図 3 のアルゴリズムは，ある部分グラフの組合せである初期解を，MDL 基準の評価値の高い順に部分グラフを選択していくという戦略を用いて生成し，計算量の許す限り組合せを変更して探索を続けていくものである．探索された組合せの各部分グラフノードとその親ノード以外は，不要なノードとしてネットワークから削除されるようになっている．図 1 では，点線で示した部分を削除し，実線で示した部分のみが残される．このアルゴリズムにより不要なノードを削除し，冗長なネットワークからグラフ集合を最も効率良く表現す

```
Algorithm SearchNext(G, Subgraphs, Previous);
begin
  Current := Previous;
  L := Length(G);
  L_prev := Length(top(Previous));
  if L < L_prev
    Current := append(G, Previous);
    Subgraphs := update(G, Subgraphs);
    if L < L_MIN then
      Minimum := Current;
      L_MIN := L;
    endif
    while Subgraphs ≠ NULL
      Next := pop(Subgraphs);
      SearchNext(Next, Subgraphs, Current);
    endwhile
  endif
end.
```

図 3: 部分グラフの組合せ探索アルゴリズム

るネットワークへ変換することができる．

2.4 柔軟なマッチングの導入による事例集合の組織化

2.3 節で説明したアルゴリズムで作成したネットワークは，完全なマッチングのみ許しているために，類似しているノードを複数作ってしまうという問題点がある．そこで，ネットワーク中のノード数を削減するために，柔軟なグラフマッチングを用いて，一つのノードにマッチするインスタンス数を増やし，類似するノードの作成を抑制する手法を検討する．

一般に，事例検索では現在の問題に完全に一致する事例が事例集合中に存在するとは限らない．したがって，最も類似した事例を検索する類似事例検索が必要となる．Bunke は，グラフ間の類似性の尺度として編集操作によるグラフ間の距離を導入し，ネットワークを利用した効率的な類似事例検索を実現している．グラフ間の距離は，節点と辺に関する置換，削除，挿入の三つの編集操作の組合せによって定義している．全てのラベルの組合せに対して各編集操作のコストが定義され，グラフ間の距離は編集操作のコストの和で定義される．SUBDUE もグラフ間の距離の定義を導入しているが，Bunke のアルゴリズムと比較して，柔軟なグラフマッチングを用いてグラフ中に瀕出するパターンの発見を行なっている点が異なっている．以上のように，類似性の定義を用いてマッチングを行なう際に

は，マッチするインスタンス数や，類似度の閾値などの何らかの基準を設けて，候補となるインスタンスを絞る必要がある．

本研究では，閾値やインスタンス数をあらかじめ設定するのではなく，その不完全なマッチングを許すことによる記述長の変化でマッチングを評価するようにしている．すなわち，不完全なインスタンスと完全なインスタンスとの間の編集操作列を表現するための記述長を，そのノードを用いた場合のグラフ集合の記述長に加え，もしインスタンス数の増加による記述長の減少量が不完全なインスタンスのマッチングによる記述長の増加量を上回るのであれば，そのマッチングは全体としての効率を上げるので，不完全なマッチングを許可するようにしている．

以上の考え方に基づいて，図2のアルゴリズムを，新しいノードにマッチするインスタンスを検索する際に，不完全なインスタンスのマッチングも行なうように拡張し，不完全なインスタンスのマッチングによる記述長の増加量について検討する．

まず，図2のアルゴリズムで，グラフ間の距離の閾値を0として，完全なマッチングからスタートし，徐々に閾値を上げていくように拡張する．閾値を上げていくと，マッチするインスタンス数は増加し，グラフ集合の記述長が減少するが，グラフ間の距離が大きくなるために編集操作に関する記述長が増加する．したがって，全体の記述長が増加し始めたら，不完全なマッチングを終了するように拡張する．このような拡張により，不完全なインスタンスもマッチングできるようになる．

次に，不完全なインスタンスに対するマッチングの記述長について検討する．ある部分グラフノード G を用いて，別のグラフ G' を表現することを考える．G から G' へのグラフの編集操作列が i_v 回の節点の挿入，i_e 回の辺の挿入，d_v 回の節点の削除，d_e 回の辺の削除，s_v 回の節点の置換，s_e 回の辺の置換から成るとする．グラフ G の節点数を v，辺の数を e，ラベル数をそれぞれ l_v, l_e とする．節点の挿入と削除を空のラベルを持つ節点との置換と考えると，$v+1$ 個の節点のうちのどの節点を l_v+1 個のラベルのどのラベルで置換するのかを記述する必要がある．組合せは $(v+1)\times(l_v+1)$ 通りなので，

$$DL_v(G \rightarrow G') = log_2(n) + \sum_{i=1}^{n} W_i(log_2(v+1) + log_2(l_v+1)) \quad (3)$$

が節点に関する操作の記述長となる．ただし，$n = i_v + d_v + s_v$ であり，W_i は各ラベルの置換に関する重みである．重みは，類似事例検索時に用いられる編集操作のコストであり，ユーザ定義による概念階層を用いて決定される．

また，辺に関してはどの二つの節点間の辺のどのラベルを l_e 個のラベルのうちのどのラベルで置換するか記述する必要がある．したがって，

$$DL_e(G \rightarrow G') = log_2 m + \sum_{i=1}^{m} W_i(2log_2 v + 2log_2(l_e+1)) \quad (4)$$

を辺に関する編集操作の記述長とする．ただし，$m = i_e + d_e + s_e$ である．

このアルゴリズムの拡張と不完全なインスタンスのマッチングに関する記述長の定義により，ノードにマッチするインスタンス数が増加し，類似するノードが一つにまとめられるために，図3のアルゴリズムで作成されるネットワークのノード数は減少する．このネットワークのノード数の減少により，事例の記憶効率と検索効率をともに向上させることができる．

3 応用例

本章では，本アルゴリズムの応用例として，人間の行動や出来事などのエピソードを記録した動画像を対象とした事例集合に適用することを検討する．一つの動画像を一つの事例とし，その内容をSchankの提案した自然言語の意味表現形式の一つである概念依存(Conceptual Dependency, 以下CDと略す)表現を用いて記述する．行為や状態などの概念を節点で，概念間の関係を辺で表現し，図4のようなラベル付き有向グラフへCD表現を変換している．なお，概念間の関係には，CD表現で定義されている数種類の因果関係の他に，after, beforeなどのAllenによる時間区間の13種の関係を用いている[6]．時間関係の処理については，本稿では紙面の都合上省略する．

事例集合を実際に組織化し，階層構造に採用された部分グラフを用いて，元の事例を記述した例を図4に

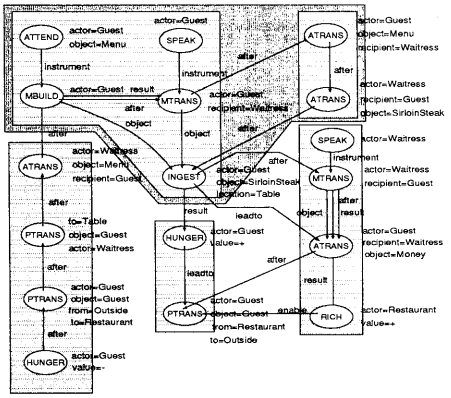

図 4: 結果の一例

示す．図 4 の斜線部分が，発見された部分グラフを示している．図 4 から分かるように，一つの事例はいくつかの部分グラフにより表現され，その部分グラフもさらに小さな部分グラフにより表現されている．すなわち，一連の出来事からなるエピソードがいくつかの場面や出来事 (行為や状態) により構成され，その場面はさらにいくつかの場面や出来事から構成されている．このような階層構造は Schank の提案した人間のエピソードに関する記憶モデル MOPs (Memory Organization Packets) に似ている [3]．以下に，MOPs の各特徴と，それを実現している本システムの機能を示す．

階層的な記憶構造 MOPs では，人間がエピソードに関する階層的な記憶構造を持つとしている．本システムでは，事例集合を階層構造で表現しており，この階層構造が MOPs の階層的な記憶構造に対応している．

過去の経験の想起 MOPs では，現在の状態に最も類似している過去の経験を想起する．この想起の過程は，事例の部分的な特徴である出来事から，類似した出来事を含むような部分グラフ (場面) を検索することに対応している．

将来の予測 MOPs では，想起した経験から，将来起こりそうな出来事や関連する場面が予測される．これは，本システムが，検索された部分グラフから，その場面に含まれる他の出来事や場面との時間関係や因果関係を用いて，関連する部分グラフを検索することに対応している．

例外的な出来事の記憶 MOPs の場面に関する記憶は，一般的な場面として典型的な一連の出来事を記憶しており，特殊な出来事はその一般的な場面の例外として記憶されている．本システムでは，事例集合を表現している階層構造中のノードが，この一般的な場面に対応している．また，各ノードは完全にマッチする事例だけではなく，完全ではないが類似している事例を，完全な事例との差により記憶している．この不完全な事例の記憶形態が，一般的な場面との相違点で記憶されている MOPs の例外に関する記憶形態に対応している．

この他に MOPs の特徴としては，新しい経験に対する動的な記憶構造の変更が上げられるが，本システムは事例の逐次的な入力に対する記憶構造の変更は考慮しておらず，これは今後の課題である．

4 おわりに

本稿は，グラフ表現された事例集合の MDL 基準による組織化手法を提案した．今後は，実際の比較的大規模な事例集合に対する組織化の実験を行ない評価を行なう予定である．

参考文献

[1] Sanders, K. E. and Kettler, B. P. and Hendler, J. A.: The Case for Structure-Based Representations., *Proceedings of the First International Conference on Case-Based Reasoning*, (1995).

[2] Messmer, B. T. and Bunke, H.: A Network Based Approach to Exact and Inexact Graph Matching. Technical Report, IAM-93-021, University of Berne (1993).

[3] Schank, R. C.: Dynamic Memory, Cambridge University Press (1982), 黒川利明, 黒川容子 (共訳), ダイナミック・メモリ, 近代科学社 (1988).

[4] 山西健司: MDL 入門: 計算論的学習理論の立場から, 人工知能学会誌, vol. 7, No. 3, pp. 435–442 (1992).

[5] Cook, D. J. and Holder, L. B.: Substructure Discovery Using Minimumu Description Length and Background Knowledge, Journal of Artificial Intelligence Research, vol. 1, pp. 231–255 (1994).

[6] Allen, J. F. and Ferguson, G.: Actions and Events in Interval Temporal Logic, Technical Report, 521, University of Rochester (1994).