数理解析研究所講究録
第790巻 1992年 29-35

計算データからの論理回路の構成

下薗 真一

Shinichi Shimozono

九州大学 大学院 総合理工学研究科 情報システム学専攻

要旨

任意の入力を与え, その出力を得ることはできるが, その構造を知ることができない未知の論理回路が与えられたとしよう. このとき, これと同じ計算を行なう論理回路を構成するにはどのような条件が必要になるのか, また計算量はどの程度になるのだろうか. 本論文ではまずこの問題について定義を与え, 構成アルゴリズムが多項式時間で出力できるような論理回路の性質を考える. そして and および or ゲートの値をすべて出力する二進木状の論理回路について, 多項式時間構成アルゴリズムが存在することを示す.

1 はじめに

正しい計算を行なう計算機をできるだけ計算方法の情報なしに, たとえば模範となる複数の入力と出力の対から構成しようという問題が, さまざまな枠組で現在研究されている. その多くは機械学習と呼ばれ, あたえる情報とそのプロトコルに制限を加えた枠組を定義し, そのとき“学習可能な”計算クラスの解明を行なっている [7]. たとえば, 帰納推論 [1], MAT 学習 [2], PAC 学習 [9] などである.

またグラフのように構造を持つものに対して計算を定義し, データをその計算結果としてあたえ, その出力をする構造を特定するという問題を扱ったものもある. たとえば, Gold は Mealy モデルのオートマトンを取り上げ, 文字列とその出力の組をあたえたとき, 最小状態数のオートマトンを構成する問題は NP-hard であることを示した [6]. また, 辺に文字を割り当てられたグラフの最小のものを, 文字列 (の集合) から推測する問題も取り上げられている [3, 8].

本論文では, 計算構造の細部を知ることはできないが, 入力と出力が定義され, 任意に入力しその出力を得ることができる未知論理回路 X が与えられたときに,X と同じ計算を行なう論理回路を構成できるか, という問題を考える. そしてそのような論理回路の性質として二進木状の論理回路の定義をあたえる.

2 論理回路とその同定問題

まず問題を次のように定義しよう.

定義 1. 未知論理回路構成問題.
入力の集合が $\{0,1\}^n$, 出力の集合が $\{0,1\}^m$ であるような論理回路の集合 $M(n,m)$ と未知論理回路 $X \in M(n,m)$ が与えられ, X に任意の入力 $x \in \{0,1\}^n$ に対する出力 $y \in \{0,1\}^m$ を計算させることができるとする.
このとき, 任意の入力について X と同じ出力をする論理回路 $c \in M(n,m)$ を構成せよ.

ここで, 同じ入力の集合および出力の集合を持つ 2 つの論理回路 (計算機) が任意の入力に対して同じ出力をするとき, それらは同一であるという.

この問題が, 決定性アルゴリズム A によって n の多項式時間で計算可能な場合はどのようなものか考えよう. まず, A は X の計算時間を無視したとしても高々多項式回の計算ステップで停止しなければならない. さらに, インスタンス $\Pi = (M(n,m), X)$ があたえられたとき $|M(n,m)| = 1$ でない限り X の計算は少なくとも 1 度は行なわれるから, すべての $c \in M(n,m)$ がすべての入力に対し高々 t ステップで計算可能なとき, $t, m < p(n)$ を満たす多項式 $p(n)$ が存在する. したがって $C(n,m,p)$ を $p(n)$ ゲートからなる n 入力 m 出力の論理回路 (boolean circuit) の集合とすると $M(n,m) \subseteq C(n,m,p)$ となる.

しかし, $p(n)$ ゲートからなる n 入力 m 出力の論理回路の数 $|C(n,m,p)|$ は n の多項式をはるかに上回るから, X を多項式回呼び出すだけで同一な計算機を構成するのは無理である. そこで次のように, 一般的な論理回路よりもその構造の決定が容易であると思われる, and および or ゲートの値すべてを出力とする論理回路の属を考えよう.

定義 2. 全出力論理回路.
n 入力の論理回路 c が入力ゲート $x_1, \ldots, x_n$ と and ゲート, or ゲートまたは 1 出力の not ゲート $g_1, \ldots, g_N$ からなり, 任意のゲート g_j について g_i がその入力となっているとき, $i < j$ が成り立つとする. このとき, c がすべての and ゲートと or ゲート $g_{c_1}, g_{c_2}, \ldots, g_{c_m}$ (ただし $c_1 < c_2 < \cdots < c_m$) の値を出力とするならば, c を全出力論理回路という.

この定義から, 未知論理回路の構成問題において $M(n,m)$ が n 入力 m 出力の全出力論理回路の集合であれば, 各論理ゲートの接続および not ゲートの有無を出力値に合うよう決定し同一な論理回路を構成できることがわかる. これは, たとえば出力が最終ゲートに限られるような回路を扱うのに比較すれば簡単であると思われる. しかし, この全出力論理回路についても次の問題 1(の補問題) が NP 完全となることから, 同一な回路を見つける多項式時間アルゴリズムが存在しそうにないことが示される.

問題 1. 論理回路の同定問題 (Boolean circuit identification, BCI).
n 入力 m 出力の全出力論理回路の対 (X,c) が与えられたとする. ただし X は未知論理回路とし

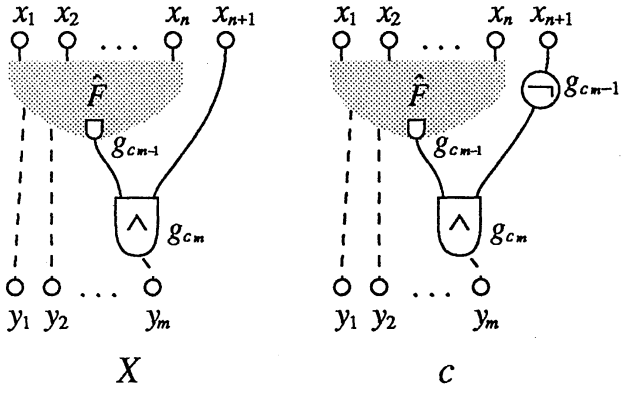

図 1: $F(x_1,\ldots,x_n)$ に対して構成される BCI のインスタンス X と c.

てその構造を知ることができないとする.
このとき, X と c は同一か?

まず, X と c の出力が一致しない入力を見つけることができれば同一でないことが多項式時間で示せるから, BCI $\in$ co-NP はすぐにわかる. さらに $\overline{\text{BCI}}$ (BCI の補問題) が NP 完全であることを証明しよう.

定理 1. $\overline{\text{BCI}}$ は NP 完全である.

証明. 任意の SAT のインスタンス $F(x_1,\ldots,x_n)$ が与えられたとき, つぎのように BCI のインスタンス (X,c) を構成する.

X と c は $n+1$ 入力 m 出力の全出力論理回路で, どちらも F を and ゲートと or ゲート $m-1$ 個およびいくつかの not ゲートで表現した部分論理回路 $\hat{F}$ をもつ. さらにどちらも出力 y_m の値を計算する and ゲート g_{c_m} を持つが, 唯一 g_{c_m} が x_{n+1} を入力とするか $\neg x_{n+1}$ を入力とするかが異なる.

まず $\hat{F}$ を構成するには, F の各リテラル $x_i, \bar{x}_i$ のかわりに x_i とこれに not ゲートを接続したものを用い, さらにすべての and と or の演算を and および or ゲートで置き換える. これは n の多項式程度の個数の and および or ゲートで可能である. ここで $\hat{F}$ の最終ゲート $g_{c_{m-1}}$ は F の値を出力することに注意する. そして X には $g_{c_{m-1}}$ と x_{n+1} を入力とする and ゲート g_{c_m} を, c には $g_{c_{m-1}}$ と $g_{c_m-1} = \neg(x_{n+1})$ を入力とする and ゲート g_{c_m} を加える (図 1).

以上の作業は n の多項式時間で終了する.

つぎに, F の充足代入があるとき, そのときに限り $(X,c) \notin$ BCI を示す入力があることを示そう.

($\Rightarrow$) 変数代入 $b_1,\ldots,b_n$ によって F が充足されるときに限り, $b_1,\ldots,b_n,1$ を X および c に入力すると $y_{m-1}=g_{c_{m-1}}$ の値が両方で 1 となる. したがって $y_m=g_{c_m}$ の値が異なり, $(X,c)\notin$ BCI がわかる.
($\Leftarrow$) 回路への入力 $b_1,\ldots,b_n,b_{n+1}$ によって X と c が同一でないことがわかるとすれば, その構成方法より $y_m=g_{c_m}$ の値の不一致によることは明らかである. g_{c_m} は and ゲートであるから, その値が異なるには X,c のどちらか一方で $g_{c_{m-1}},g_{c_m-1}$ 両方の値が 1 でなければならない. このとき $g_{c_{m-1}}$ の値は F に $b_1,\ldots,b_n$ を入力したときの値であるから, これは F の充足代入である. したがって F が充足可能であることがわかる. □

この結果から, n 入力 m 出力 ($m<p(n)$, $p(n)$ は n の多項式) の全出力論理回路で記述可能な X が与えられたとき, P $\neq$ NP の仮定のもとで, これに同一な論理回路をみつける決定性の多項式時間アルゴリズムは存在しないことがわかる.

3 多項式時間で構成可能な論理回路

前節で未知論理回路の構成問題は, $M(n,m)$ が n の多項式時間で計算可能な全出力論理回路の集合の場合でも, n の多項式時間でできそうにないことがわかった. そこで多項式時間で構成可能な論理回路の性質として次のようなものを考える.

定義 3. 二進木状論理回路.
ループバックのない論理回路 c の各ゲートの出力線 (fan out) が高々 1 であるならば, c は二進木状であるという.

この性質をみたす n 入力の論理回路の and, or ゲートの個数 m は高々 $n-1$ 個である. また, 論理回路中のある論理ゲートに対する部分回路を次のように定義する.

定義 4. 関連ゲートと部分回路.
c を n 入力, N 論理ゲートの二進木状の論理回路とする. c の論理ゲート g_k $(1\leq k\leq N)$ の関連ゲートを帰納的に

i) g_k の入力となっているゲート

ii) g_k の関連ゲートの入力となっているゲート

と定義する. このとき, g_k の関連ゲートすべてからなる部分的な論理回路を, g_k の部分回路という.

すると, 二進木状論理回路の部分回路について次のような性質があることがわかる.

補題 1. g_k を n 入力の二進木状論理回路 c の任意の論理ゲートとすると, その値は g_k の部分回路に含まれる入力ゲートの値から一意に決まる. さらに g_k の値が 0 または 1 となる c の入力は $\mathcal{O}(n)$

で計算可能である. また, 任意の論理ゲート g_i $(1 \leq i < k)$ の関連ゲートが1つでも g_k の関連ゲートならば, g_i は g_k の関連ゲートであり, g_i の部分回路はすべて g_k の部分回路に含まれる.

これらの性質を用いて, 次のような結果が得られた.

定理 2. 未知論理回路 $X \in M(n,m)$ が与えられたとき, $M(n,m)$ が n 入力 m 出力の二進木状全出力論理回路の集合ならば, $\mathcal{O}(n^4)$ 時間で X に同一な論理回路 $c \in M(n,m)$ を構成できる.

証明. ゲート g_{c_k} を y_k で, また g_{c_k} に接続した not ゲートを $\neg y_k$ で表すとし, n 入力 m 出力の, 二進木状の全出力論理回路 c を $\{y_1,\ldots,y_m\}$ から $\{\mathrm{and},\mathrm{or}\} \times \{x_1,\neg x_1,\ldots,x_n,\neg x_n, y_1,\neg y_1,\ldots,y_m,\neg y_m\}^2$ への関数 Γ で表現する. また X に $b \in \{0,1\}^n$ を入力したときの出力の k ビット目を $X_k(b)$ と書くとする.

このとき, 次のような algorithm で X に同一な論理回路 c を見つけることができる.

Algorithm

すべての $\gamma_k \in \{y_1,\ldots,y_m\}$ について $\Gamma(\gamma_k)$ を未定義とする.

入力として利用可能なゲートの集合 G を $G = \{x_1,\ldots,x_n\}$ とする.

(1) $1 \leq k \leq m$ なる y_k に対して順に:

(2) $\{(\gamma_i,\gamma_j) \mid \gamma_i \neq \gamma_j \in G\}$ すべてについて:

γ_i と γ_j の関連ゲートに含まれる入力ゲート以外の入力ゲートの値がすべて一致し, (γ_i, γ_j) の値が $(0,0)$, $(0,1)$, $(1,0)$, $(1,1)$ とるような入力 $b_{ij}(0)$, $b_{ij}(1)$, $b_{ij}(2)$, $b_{ij}(3)$ を Γ から計算する. X に $b_{ij}(0),\ldots,b_{ij}(3)$ を入力し, それぞれについての k ビット目の出力の値 X_k を調べる. もし $X_k(x_{ij}(0)),\ldots,X_k(x_{ij}(3))$ の値が

$0,0,0,1$ ならば $\Gamma(y_k) = (\mathrm{and},\gamma_i,\gamma_j)$;
$0,1,1,1$ ならば $\Gamma(y_k) = (\mathrm{or},\gamma_i,\gamma_j)$;
$0,1,0,0$ ならば $\Gamma(y_k) = (\mathrm{and},\neg\gamma_i,\gamma_j)$;
$0,0,1,0$ ならば $\Gamma(y_k) = (\mathrm{and},\gamma_i,\neg\gamma_j)$;
$1,1,0,1$ ならば $\Gamma(y_k) = (\mathrm{or},\neg\gamma_i,\gamma_j)$;
$1,0,1,1$ ならば $\Gamma(y_k) = (\mathrm{or},\gamma_i,\neg\gamma_j)$;
$1,0,0,0$ ならば $\Gamma(y_k) = (\mathrm{and},\neg\gamma_i,\neg\gamma_j)$;
$1,1,1,0$ ならば, $\Gamma(y_k) = (\mathrm{or},\neg\gamma_i,\neg\gamma_j)$;

と y_k の論理ゲートの種類と入力を決定し, (2) を終了する.

G から γ_i,γ_j を除き, y_k をくわえる.

c を出力し停止する.

ではこの algorithm が X と同一の c をみつけて停止することを帰納的に証明しよう.

Base step: まず algorithm は y_1 の入力対を $\{(\gamma_i,\gamma_j) \mid \gamma_i \neq \gamma_j \in G\}$ の中から選ぶが, このとき $G=\{x_1,\ldots,x_n\}$ で, algorithm は組合せのすべてを尽くす $(\mathcal{O}(n^2))$ から, X における y_k の入力と同じゲートの組合せを見つけ出す. このとき, 二進木状の論理回路の定義より, not ゲートの有無と論理演算の種類の組合せは algorithm で検証する 8 種類しかないから, 出力パターン $X_k(b_{ij}(0)),\ldots,X_k(b_{ij}(3))$ がどれかと一致する. したがって algorithm は not ゲート g_{c_1-1} の有無 (すなわち c_1 の番号), $y_1=g_{c_1}$ の種類および入力を決定する.

ここで決定された y_1 の入力が間違っていたとしよう. すると algorithm で得た g_{c_1} の入力 γ_i,γ_j は X のそれと少なくとも 1 つが異なる. したがって, X に γ_i,γ_j 以外の G に含まれるゲートの値が変化せず, (γ_i,γ_j) の値が (0,0), (0,1), (1,0), (1,1) となるような $b_{ij}(0)$, $b_{ij}(1)$, $b_{ij}(2)$, $b_{ij}(3)$ を Γ から計算して入力すると, $X_k(x_{ij}(0)),\ldots,X_k(x_{ij}(3))$ はどのような場合でも, 少なくとも 1 つは c で得られる y_1 の値, すなわち algorithm で照合を行なう真偽値と異なる. しかし algorithm ではこのような接続は許していないから, これは仮定に反する. したがって c における $y_1=g_{c_1}$ の入力および演算の種類は X と同じである.

Induction step: c のゲートの入力接続およびその種類が y_{k-1} まで決まっており, しかも X と同じであるとする.

このとき $y_k=g_{c_k}$ の入力として接続可能なゲートは論理回路の性質より G に含まれるゲートのみであり, $|G|<n$ である. したがって base step での場合と同じように, algorithm はすべてを尽くすことで必ず X の $y_k=g_{c_k}$ と一致する接続と種類をみつけ, その決定を $\mathcal{O}(n^3)$ 時間で終了する.

ここで決定された y_k の入力接続が間違っていたとしよう. y_k の入力として接続可能なゲートは, 1 度も入力として使用されていない G に含まれるゲートに限られ, しかも補題 1 によりそれぞれの関連ゲートは全く重複しないから, 回路の入力として γ_i,γ_j 以外の接続可能なゲートの値が変化せず, (γ_i,γ_j) の値が (0,0), (0,1), (1,0), (1,1) となるような入力 $b_{ij}(0), b_{ij}(1), b_{ij}(2), b_{ij}(3)$ が $\mathcal{O}(n)$ で計算できる. 仮定により c の $y_k=g_{c_k}$ の入力 γ_i,γ_j は, X のそれと少なくとも 1 つは異なっているから, base step の場合と同じように, c の出力 y_k と $X_k(x_{ij}(0)),\ldots,X_k(x_{ij}(3))$ を照合するとどのような場合でも 1 つは異なる. しかし algorithm はこのような接続を行なわないから, これは仮定に反する. したがって c の y_k の接続は X と同じである.

以上の議論により, algorithm は $y_1=g_{c_1},\ldots,y_m=g_{c_m}$ について X と同じ接続を $\mathcal{O}(n^4)$ 時間で探索することが証明される. □

4 おわりに

本論文では, 多項式時間で未知論理回路から構成できる論理回路の性質をあたえた. これは not ゲートは使用してよいが, and, or ゲートの値をすべて出力し, しかも二進木構造しか許されない,

非常に制約の多いものである. 一方, ここで扱った問題と同じように入力を与えて出力を得ることを繰り返し, $\{0,1\}^n$ から $\{0,1\}$ への論理関数を同定する問題がすでに研究されており, 制約を持つ単調 (monotonic) な論理関数に対する多項式時間アルゴリズムが示されている [4]. このような結果を踏まえて, ゲートの fan out が 2 以上であるような論理回路の性質について, 調べる必要がある.

また一般の論理回路, すなわちすべての多項式時間で計算可能な論理回路については, 多項式時間アルゴリズムが存在しそうにないことを示したのみである. $M(n,m)$ を n の多項式時間計算機の集合としたとき, 厳密にどの程度の計算量が必要なのか, また, 近似的な枠組やアルゴリズムを考えた場合どのようになるかは今後の研究課題である.

参考文献

[1] Angluin, D. and Smith, C. H., Inductive inference : Theory and methods, *Computing Surveys*, **15**, 237–269, 1983.

[2] Angluin, D., Learning regular sets from queries and counter-examples, Yaleu/dcs/tr-464, Yale University, Department of Computer Science, 1986.

[3] Aslam, J. A. and Rivest, R. L., Inferring graphs from walks, In *Proceedings of the 3rd Annual Workshop on Computational Learning Theory*, pages 359–370, 1990.

[4] Boros, E., Hammer, P. L., Ibaraki, T. and Kawakami, K., Identifying 2-monotonic positive boolean function in polynomial time, In *Lecture Notes on Computer Science*, 1991.

[5] Garey, M. R. and Johnson, D. S., *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W. H. Freeman and Company, 1978.

[6] Gold, E. M., Complexity of automaton identification from given data, *Information and Control*, **37**, 302–320, 1978.

[7] Laird, P., A survey of computational learning theory, NASA ames research center, artificial intelligence research branch, Survey, 1989.

[8] Maruyama, O. and Miyano, S., Inferring a tree from walks, Technical report, Research Institute of Fundamental Information Science, Kyushu University, 1991.

[9] Valiant, L. G., A theory of the learnable, *Communication of the ACM*, **27** (II), 1984.