

実閉体上の非冠頭標準形論理式に対する限量記号消去に おける部分論理式の処理順序と計算効率

Efficient Subformula Orders for Real Quantifier Elimination of Non-prenex Formulas

小林宗広*

MUNEHIRO KOBAYASHI

筑波大学大学院数理物質科学研究科

GRADUATE SCHOOL OF PURE AND APPLIED SCIENCES, UNIVERSITY OF TSUKUBA

岩根秀直†

(株) 富士通研究所 / 国立情報学研究所

FUJITSU LABORATORIES LTD. / NATIONAL INSTITUTE OF INFORMATICS

Abstract

本講究録では実閉体における非冠頭標準形の論理式の限量記号消去 (quantifier elimination, QE) アルゴリズムの効率化について議論する。非冠頭標準形の一階述語論理式の QE 問題を部分論理式の QE 問題に帰着させる際にその処理順を最適化することに注目した。部分論理式を処理する適切な順序を決定するためにヒューリスティクス及び機械学習の 2 種類の方法を提案し、2,000 問以上の非自明な問題に対する実験を通してそれらの順序関数の効果を測定した。本実験の結果により機械学習が効果的なヒューリスティクスの開発にかかる試行錯誤の手間を実行効率を犠牲にすることなしに大幅に省くことができる可能性が示唆された。

1 導入

本研究の目標は機械学習を用いてヒューリスティクス開発を自動化し、実閉体における非冠頭標準形の論理式の限量記号消去 (quantifier elimination, QE) アルゴリズムの効率化に役立てることである。

実閉体上の QE の議論では入力論理式が冠頭標準形であることを仮定している場合が多い。冠頭標準形は限量記号が論理式の先頭部分に集められている形式である。しかし、実用の場面で入力論理式が常に冠頭標準形で与えられるとは限らない。非冠頭標準形の論理式を冠頭標準形に変形することにより変数間の代数的独立性を利用しづらくなり、QE 計算の効率を損なう可能性がある。従って、実用上の観点から、非冠頭標準形の論理式を部分論理式に分割、各部分論理式に QE アルゴリズムを適用し、それぞれの結果を論理的に統合するという方針が効果的であると考えられる。この際、途中の QE 計算結果を用いて QE を未実行の残りの部分論理式を単純化することにより更なる効率化を達成できる。

*munehiro-k@math.tsukuba.ac.jp

†iwane@jp.fujitsu.com

ここで、部分論理式の QE 計算をどのような順番で実行するのか決定するという問題が発生する。入力論理式の QE 計算に必要な時間は部分論理式の順序に依存している。このような状況ではしばしば部分論理式の適切な順序を決定するために「簡単な」部分論理式を判定するヒューリスティクスが導入される。実際に [10] ではこのようなヒューリスティクスの導入により QE 計算の効率化が達成されている。

実用上有用であるにもかかわらず、非冠頭標準形の論理式に対する QE アルゴリズムの研究は少ない [2, 10]。現状では非冠頭標準形の論理式を QE の入力として受け取ることができる数式処理システムは Mathematica と Maple 上のパッケージ SyNRAC [11] のみである。本研究では実用上の QE アルゴリズムの効率を改善するために非冠頭標準形の論理式に対する QE に着目した。

数式処理において QE アルゴリズムの高速化を目的として変数順序を決定するヒューリスティクスを提案する先行研究が存在している。Dolzmann らは cylindrical algebraic decomposition (CAD) の最適な射影順序を推定するヒューリスティクスを提案した [6]。その後 Huang らは機械学習を CAD の射影順序を推定するヒューリスティクスの選択に応用した [8]。Huang らはサポートベクターマシン (support vector machine, SVM) がヒューリスティクス選択に有効であり、どのヒューリスティクスを単独で用いるよりも SVM によってヒューリスティクスを切り替えた場合の方が効率がよいことを示した。Huang らの研究が主に CAD の射影因子の数によって性能を評価していたのに対し、本研究では実行時間によって性能評価を行い、より実用に近い研究を行った。

本実験は 2,306 問の非自明な QE 問題を使用して行った。これらの QE 問題は人工知能プロジェクト「ロボットは東大に入れるか」[1, 12] の数学チームの活動から得られたものである。「ロボットは東大に入れるか」は国立情報学研究所が中心となって 2011 年に立ち上げられたプロジェクトであり、自然言語で記述された大学入試問題を自動で解答する人工知能システムを開発することを目的としている。本研究はこのプロジェクトへの貢献を目指して行われた。このため、実行時間による性能評価の中でも、単純な実行時間の長さより QE 計算が指定された時間内で終了するかどうかを重視した。

入力問題とその計算結果を機械学習を適用するために十分な量集められる場合、本実験の結果もまた、機械学習の応用により有用なヒューリスティクスが得られることを示している。有用なヒューリスティクス開発は多くの試行錯誤を伴うため、機械学習の応用がヒューリスティクス開発の効率化にとって有効な選択肢となりうることを示唆された。

2 対象とする問題

この講究録では実閉体の非冠頭標準形の論理式に対する QE の高速化を扱う。冠頭標準形は限量記号が論理式の先頭部分に集められている形式であり、非冠頭標準形の論理式の実例として次のような論理式 φ が挙げられる：

$$\varphi \equiv \varphi_1 \vee \varphi_2 \vee \varphi_3, \quad (1)$$

ここで、

$$\begin{aligned} \varphi_1 &\equiv \exists x_0 (-x_0 \leq -1), \\ \varphi_2 &\equiv \exists x_1 (x_1 \leq 0), \\ \varphi_3 &\equiv \exists x_2 \exists x_3 \exists x_4 \left(\exists x_5 ((x_2^3 x_4 x_5 - x_2^2 x_4 - x_2^2 x_5 + x_2 + 1 = 0 \vee \right. \\ &\quad x_2^3 x_4 x_5 x_6 - x_2^2 x_4 x_5 - x_2^2 x_4 x_6 - x_2^2 x_5 x_6 + x_2 x_4 + x_2 x_5 + x_2 x_6 - x_6 = 0) \wedge \\ &\quad \left. (x_2^3 x_4 x_6 - x_2^2 x_4 - x_2^2 x_6 + x_2 + 1 = 0 \vee \right. \end{aligned}$$

$$\begin{aligned}
& (x_2^3 x_4 x_5 x_6 - x_2^2 x_4 x_5 - x_2^2 x_4 x_6 - x_2^2 x_5 x_6 + x_2 x_4 + x_2 x_5 + x_2 x_6 - x_5 = 0) \wedge \\
& (x_2^3 x_3 - x_2^2 x_3 - x_2^2 + x_2 + 1 = 0 \vee \\
& x_2^3 x_3 x_4 - x_2^2 x_3 x_4 - x_2^2 x_3 - x_2^2 x_4 + x_2 x_3 + x_2 x_4 + x_2 - x_4 = 0) \wedge \\
& (x_2^3 x_4 - x_2^2 x_4 - x_2^2 + x_2 + 1 = 0 \vee \\
& x_2^3 x_3 x_4 - x_2^2 x_3 x_4 - x_2^2 x_3 - x_2^2 x_4 + x_2 x_3 + x_2 x_4 + x_2 - x_3 = 0) \Big).
\end{aligned}$$

部分論理式を処理する順序を入れ替えることにより、QE 計算の高速化を図るのが基本的なアイデアである。

岩根らの論文 [10] に、部分論理式を処理する順序が QE 計算の効率に影響を与えることが報告されている。そこでは論理式が機械的に生成された場合など入力冗長である場合、専用 QE アルゴリズムの利用、途中式の簡単化、より簡単な部分論理式への分解が効率的な QE 計算に効果的であると述べられている。Algorithm 1 ([10] より引用) に岩根らの提案する非冠頭標準形の論理式に対する QE アルゴリズムを示す。Algorithm 1 中の $\text{QE}_{\text{prenex}}$ は CAD, Virtual Substitution, $\forall x(f(x) > 0)$ 及び $\forall x(x \geq 0 \rightarrow f(x) > 0)$ の形を持つ論理式に対する実根の数え上げによる QE [9] などの既存のアルゴリズムを実行する。このアルゴリズムでは入力を非冠頭標準形の論理式に変形、部分論理式をソートし、各部分論理式に対し QE を再帰的に実行する。また、先に QE 計算が実行されて得られた結果は以降の部分論理式を簡単化するための条件として用いられる。このため、部分論理式は QE 計算が簡単なものから難しい順序に並んでいる場合に効率がよいと考えられる。今回の実験では Algorithm 1 中の ‘SORT’ を決定する順序関数を変更することによるアルゴリズムの実行速度への影響を測定した。

Algorithm 1 $\text{QE}_{\text{main}}(\varphi, \psi_{\text{nec}}, \psi_{\text{suf}})$

Input: a first-order formula φ , quantifier-free formulas ψ_{nec} and ψ_{suf}

Output: a quantifier-free formula φ' s.t. $\varphi \wedge \psi_{\text{nec}} \vee \psi_{\text{suf}}$ is equivalent to $\varphi' \wedge \psi_{\text{nec}} \vee \psi_{\text{suf}}$

```

1: if  $\varphi$  is quantifier-free then
2:   return SIMPLIFY( $\varphi, \psi_{\text{nec}}, \psi_{\text{suf}}$ )
3: else if  $\varphi$  is a prenex formula then
4:   return  $\text{QE}_{\text{prenex}}(\varphi, \psi_{\text{nec}}, \psi_{\text{suf}})$ 
5: else if  $\varphi \equiv Q_1 x_1 \cdots Q_m x_m \xi$  where  $Q_j \in \{\exists, \forall\}$  then
6:   /*  $\xi$  is not quantifier-free */
7:   return  $\text{QE}_{\text{prenex}}(Q_1 x_1 \cdots Q_m x_m \text{QE}_{\text{main}}(\xi, \psi_{\text{nec}}, \psi_{\text{suf}}), \psi_{\text{nec}}, \psi_{\text{suf}})$ 
8: else if  $\varphi \equiv \vee \xi_i$  then
9:   return  $\neg \text{QE}_{\text{main}}(\neg \varphi, \neg \psi_{\text{suf}}, \neg \psi_{\text{nec}})$ 
10: /*  $\varphi \equiv \wedge \xi_i$  */
11: if  $\xi_i \equiv (f_i = 0)$  and  $f_i$  is reducible ( $f_i = \prod g_{i,j}$ ) then
12:   return  $\text{QE}_{\text{main}}(\vee_j (g_{i,j} = 0 \wedge (\wedge_{k \neq i} \xi_k)), \psi_{\text{nec}}, \psi_{\text{suf}})$ 
13:  $\{\varphi_i\}_{i=1}^n \leftarrow \text{SORT}(\{\xi_i\}_{i=1}^n)$ 
14: for all  $i$  such that  $1 \leq i \leq n$  do
15:    $\varphi_i \leftarrow \text{QE}_{\text{main}}(\varphi_i, \psi_{\text{nec}}, \psi_{\text{suf}})$ 
16:    $\psi_{\text{nec}} \leftarrow \psi_{\text{nec}} \wedge \varphi_i$ 
17: return  $\wedge_i \varphi_i$ 

```

3 諸概念

ここでは、実験に用いた順序関数を説明する。今回使用した順序関数には大きく分けてヒューリスティクス・ランダム・機械学習の3種類がある。機械学習を利用した順序関数の設定に関してもここで示す。

3.1 順序関数

以下の論理式に関する順序関数を設定した：

ヒューリスティクス 以下の特徴量が論理式から計算され、その量を比較することで論理式の順序を決定する。本実験では10個のヒューリスティクスを設定した。

ADG [10]. 特徴量は表1の1, 2, 3, 19, 27, 45の重み付き和である。

nvar, npoly, sotd, msotd, mtdeg, mdeg, and mterm. 特徴量はそれぞれ表1の1, 2, 3, 7, 8, 9, 10である。

nvar_rev and npoly_rev. 特徴量はそれぞれ表1の1, 2である。これらの順序関数はそれぞれ‘nvar’と‘npoly’の逆順序であり、順序関数がQE計算の効率に与える悪影響を調べる目的で用意された。

ランダム 各部分論理式に乱数で優先度を割り振る。

機械学習 事前に訓練されたモデルを用いてSVMにより論理式のdecision valueを計算する。decision valueが大きい論理式がより小さい論理式であると評価される。decision valueについては3.2節において説明する。

上記の設定から分かるように、ヒューリスティクス及び機械学習の順序関数はQE計算において部分論理式の最適な順序を与えるような表1中の特徴量の組み合わせを探るという点で同じである。

3.2 機械学習による順序関数の設定

3.2.1 特徴量

論理式にSVMを適用するためには、固定長の数値ベクトルにより論理式を特徴付ける必要がある。特徴ベクトルの各特徴量は論理式のなんらかの特徴を捉えていることが期待される。今回そのような特徴ベクトルを構成するために表1に挙げる番号1から58の特徴量を用いた。これらの特徴量は先行研究[6, 3, 8]及び専用QEアルゴリズムの適用条件から採用した。

表1で‘特徴量4’が‘特徴量3’を‘特徴量2’で割ったもので表現できるなど、他の特徴量の組み合わせで表現できるような特徴量も含まれているが、このような従属な特徴量は特徴量の間の非線形な関係を強調する目的で選ばれている。radial basis function (RBF) カーネルを用いたSVMでは各特徴量やそれらの関係に対して偏りのない反応性を持っているが、特に重要であることが分かっている特徴量の関係がある場合、その関係をよく反映するような従属な特徴量を加えることによりSVMの性質を変えることができる。

また、学習対象に対して適切なカーネルを設計することにより、SVMの性質を変えることもできる。この方針に関しては3.3節でより詳しく議論する。本実験では特に指定しなかった場合、RBFカーネルを使って学習するものとする。

3.2.2 ラベル付け

SVM を利用するためには訓練データに +1 または -1 のラベルを付する必要がある。本実験では訓練データ中の QE 問題が N 秒以下で解かれる場合に対応する特徴ベクトルに +1 のラベルを付し、それ以外の場合に -1 のラベルを付した。 N としては 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 20, 30, 55 を用いた。 閾値の 55 秒は訓練データ中で実行時間の短い 90 パーセントと実行時間の長い 10 パーセントを分離する時間である。

3.2.3 SVM を用いた順序関数の設定

SVM の機能としては一番単純な 2 クラス分類を利用して順序関数を設定した。 SVM を使って 2 クラス分類を行うには 2 ステップが必要とされる。 まず, +1 (正例) または -1 (負例) が付された特徴ベクトルからなる訓練データからモデルを生成する (training)。 その後, 得られたモデルに従って新しい論理式が正例と負例のどちらに属するかを推定する (predict)。

第 1 ステップにおいて SVM は何らかのベクトル空間において訓練データの正例と負例を分離する超平面を計算する。 第 2 ステップにおいて SVM は新しい論理式に対して decision value を計算する。 decision value は第 2 ステップで入力された論理式と第 1 ステップで計算した超平面の距離の一種であり, 推定の正しさの指標と考えられる。 このため, より大きな decision value を持つ論理式から優先して QE 計算を行った。 図 1 に SVM を用いて部分論理式をソートする手順を図示する。

本実験ではラベル付けの閾値を変えることにより 13 のモデルを訓練したほか, 3.3 節で述べるカーネルを利用することにより別の 1 つのモデルを訓練した。

3.3 カーネル選択

適当なカーネルを定義することにより, 論理式を固定長のベクトル以外の構造で特徴付けることができるようになる。 カーネル設計に関しては一般的な枠組みとして畳み込みカーネル [7] が提案されており, 解説論文として [14] がある。 自然言語処理 [5] や, HTML データの分類 [15] などへの応用の報告があるが, 論理式を評価するためのカーネル設計は先行研究が少なく, どのようなカーネルが効果的かよく知られていない。

今回, 論理式の評価用にカーネルの設計を試みた。 主に実装の容易さから, 導出部分カーネル [13] と RBF カーネルの合成によるカーネルを実装した。 このカーネルの使用時において, 論理式は変数の数に依存する可変長のベクトルで表される。 表 1 の特徴量 x_i ($i = 1, \dots, 6$) が各変数に対して計算され, それらを並べたベクトルが特徴ベクトルとなる。

Definition 1. 論理式 θ に表れる変数全体を X とし, 変数 $x \in X$ に対し, x_i ($i = 1, \dots, 6$) を上記の特徴量とする。 さらに, x の特徴ベクトルを $F_{\theta,x} = (x_1, \dots, x_6)$ と書く。 ただし, θ には同じ名前の束縛変数がないものとし, 束縛変数も X に含まれるとする。

論理式 φ と ψ に表れる変数全体をそれぞれ X, Y とする。 導出部分カーネル κ_{DS} を次のように定義する:

$$\kappa_{DS}(\varphi, \psi) = \sum_{x \in X} \sum_{y \in Y} \kappa_{RBF}(F_{\varphi,x}, F_{\psi,y}).$$

ここで, κ_{RBF} は RBF カーネルを表すこととする。 すなわち, $\gamma > 0$ を RBF カーネルのパラメータ, $u, v \in \mathbb{R}^n$ として

$$\kappa_{RBF}(u, v) = \exp(-\gamma \|u - v\|^2).$$

導出部分カーネルの特徴として, 特徴ベクトルを構成する変数の順序によらず κ_{DS} の値は一意に定まる。 図 2 に導出部分カーネルと SVM を用いて部分論理式をソートする手順を図示する。 ユーザ定義のカーネルを利用するために, カーネル行列を計算するステップが増えている。

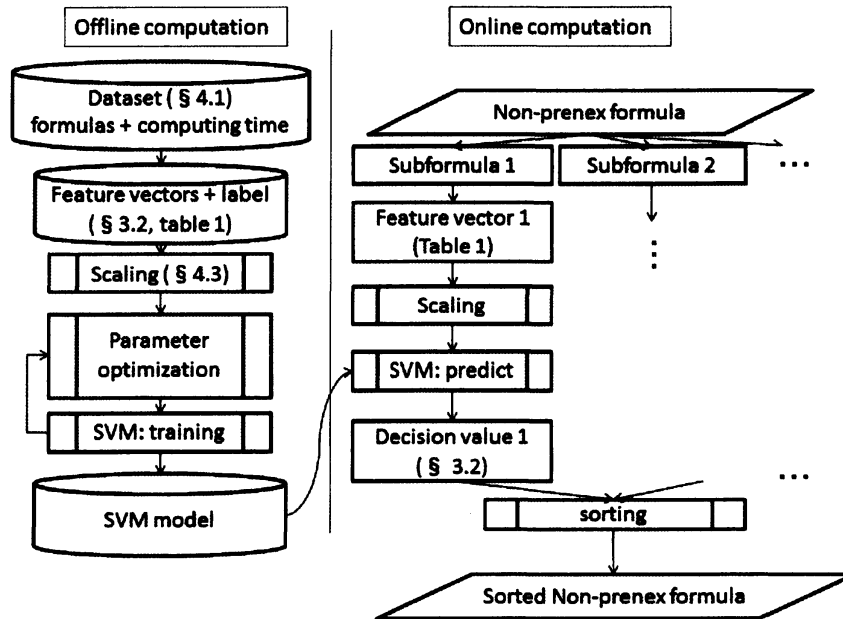


図 1: SVM を用いた部分論理式のソート (RBF カーネルを使用)

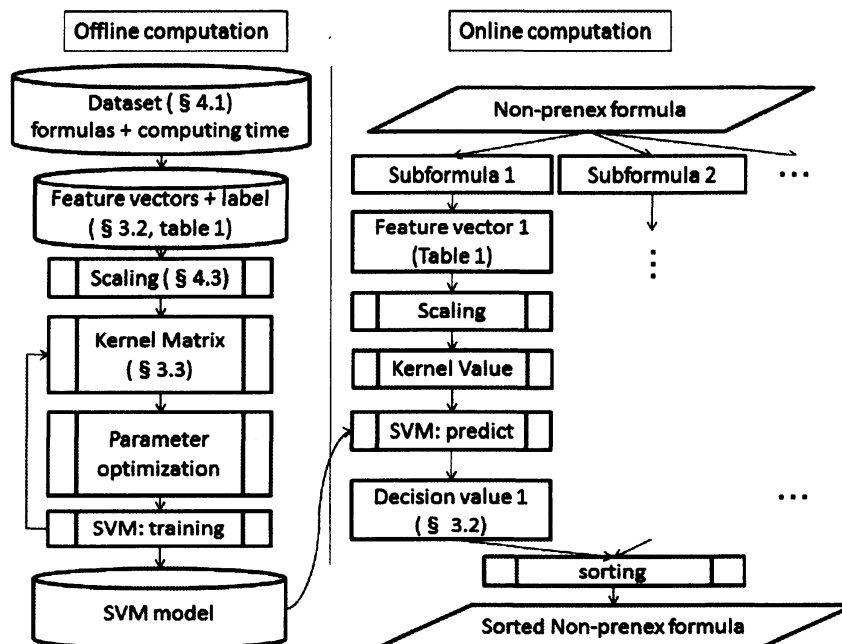


図 2: SVM を用いた部分論理式のソート (ユーザ定義カーネルを使用)

表 1: SVM による論理式の学習に用いた特徴量. 1 から 54 の特徴量は順に全ての変数, 束縛変数, 自由変数の 3 種類の変数に対して計算される. また, 論理式に表れる各変数 x に対し特徴量 $x-1$ から $x-6$ が計算される. 論理式 φ に対し, 次の記法を用いる: φ に表れる全変数, 束縛変数, 自由変数からなる集合を $\text{Var}(\varphi)$ と表記する. φ に表れる多項式 p で, $\deg_v(p) > 0$ となる $v \in \text{Var}(\varphi)$ が存在するものからなる集合を $\text{Poly}(\varphi)$ と表記する. $p \in \text{Poly}(\varphi)$ に表れる項全体からなる集合を $\text{Term}(p)$ と表記する.

Feature number	Label	Description
1, 19, 37	nvar	Number of variables
2, 20, 38	npoly	Number of polynomials
3, 21, 39	sotd	Sum of total degrees ($\sum_{p \in \text{Poly}(\varphi)} \sum_{t \in \text{Term}(p)} \sum_{v \in \text{Var}(\varphi)} \deg_v(t)$)
4, 22, 40	asotd	Average sum of total degrees w.r.t. npoly (sotd/npoly)
5, 23, 41	atdeg	Average total degree w.r.t. npoly ($\sum_{p \in \text{Poly}(\varphi)} \max_{t \in \text{Term}(p)} (\sum_{v \in \text{Var}(\varphi)} \deg_v(t)) / \text{npoly}$)
6, 24, 42	aterm	Average number of terms w.r.t. npoly ($\sum_{p \in \text{Poly}(\varphi)} \sum_{t \in \text{Term}(p)} 1 / \text{npoly}$)
7, 25, 43	msotd	Maximum sum of total degrees ($\max_{p \in \text{Poly}(\varphi)} \sum_{t \in \text{Term}(p)} \sum_{v \in \text{Var}(\varphi)} \deg_v(t)$)
8, 26, 44	mtdeg	Maximum total degree ($\max_{p \in \text{Poly}(\varphi)} \max_{t \in \text{Term}(p)} \sum_{v \in \text{Var}(\varphi)} \deg_v(t)$)
9, 27, 45	mdeg	Maximum degree ($\max_{p \in \text{Poly}(\varphi)} \max_{v \in \text{Var}(\varphi)} \deg_v(p)$)
10, 28, 46	mterm	Maximum number of terms ($\max_{p \in \text{Poly}(\varphi)} \sum_{t \in \text{Term}(p)} 1$)
11, 29, 47	ndeg1	Number of polynomials with degree 1 ($\sum_{p \in \text{Poly}(\varphi), \max_{t \in \text{Term}(p)} \sum_{v \in \text{Var}(\varphi)} \deg_v(t) = 1} 1$)
12, 30, 48	ndeg2	Number of polynomials with degree 2
13, 31, 49	ndeg3	Number of polynomials with degree 3
14, 32, 50	rdeg1	Ratio of ndeg1 and npoly (ndeg1/npoly)
15, 33, 51	rdeg2	Ratio of ndeg2 and npoly (ndeg2/npoly)
16, 34, 52	rdeg3	Ratio of ndeg3 and npoly (ndeg3/npoly)
17, 35, 53	mcoef	Maximum absolute value of coefficients ($\max_{p \in \text{Poly}(\varphi)} \max_{c \in \text{Coeff}(p)} c $)
18, 36, 54	acoef	Average absolute value of coefficients w.r.t. npoly (mcoef/npoly)
55		Ratio of the number of the symbol '=' and npoly
56		Ratio of the number of the symbol '≠' and npoly
57		Ratio of the number of the symbol '<' and npoly
58		Ratio of the number of the symbol '≤' and npoly
$x-1$		Value 0 if x is free and 1 if quantified
$x-2$		Maximum degree of x
$x-3$		Maximum total degree among the monomials containing x
$x-4$		Number of polynomials containing x
$x-5$		Number of the symbol '=' containing x in either the sides
$x-6$		Number of the symbol '≠' containing x in either the sides

4 計算実験

ここでは今回実行した計算実験の手順と結果について述べる。QE 計算には Maple 上のパッケージ SyNRAC [11] を用いた。機械学習には SVM の C++ による実装である LIBSVM [4] を用いた。全ての計算実験は Intel(R) Xeon(R) CPU E7-4870 2.40GHz メモリ 1007 GB 上で行った。

4.1 データセット

国立情報学研究所の人工知能プロジェクト「ロボットは東大に入れるか」[1, 12] によって収集された問題及び開発された数学問題解答システムより生成されたログから 2,306 個の一階述語論理式を抽出した。問題の出典はチャート式シリーズ, 旧帝大入学試験問題, 国際数学オリンピックである。この数学問題解答システムは代数, 線形代数, 幾何, 指数対数三角関数, 微積分, 組み合わせ問題, 整数問題, 高階論理などを入力として受け取ることができる。これらの問題の中から実閉体の言語で表現できるものを利用し, 今回のデータセットを作成した。主に代数, 線形代数, 幾何の問題を実閉体の言語で表現することができる。

各論理式に対し, SyNRAC により QE 計算を実行し, 実行時間を記録した。ここで, 600 秒で計算が終了しない場合にはタイムアウトとした。表 2 に 600 秒以内に計算が終了した 2,116 の論理式の計算時間の分布を示す。タイムアウトとなった論理式は 187 個あった。残りの 4 つの論理式は SyNRAC の QE 計算中にエラーを発生させた。この 4 つの論理式は機械学習の訓練データから除外した。

4.2 QE 計算の実例

2 章で見た論理式 (1) を再び考える。SVM を用いた順序関数を用いて QE 計算を行う手順を見る。 φ を QE する問題は 3 つの QE 部分問題 $\varphi_1, \varphi_2, \varphi_3$ に分けられる。この例の場合, φ_1 と φ_2 が明らかに真であるため, φ は真となる。部分問題を処理する順序は LIBSVM により計算される decision value によって決定される。表 3 に $\varphi_1, \varphi_2, \varphi_3$ の QE にかかる計算時間と特徴ベクトルを示す。また, 表 4 に QE 部分問題の decision value と処理順序を示す。順序関数 ' <2 ', ' <3 ', ' <7 ' を用いた QE 計算では φ_3 が最初に計算され, 600 秒以内に計算が終了しなかった。一方で φ_1 または φ_2 から計算を開始する順序関数を用いた QE 計算は 1 秒未満で計算が終了した。

4.3 実験結果

本実験では 25 個の順序関数を用意した。これらの内, 10 個のヒューリスティクスと 1 個のランダムに関しては 3.1 節において述べた通りである。残りの 14 個は機械学習を用いて設定された順序関数である。3.2 節において述べた RBF カーネルを用いた機械学習を利用した 13 個の順序関数はラベル付けの閾値によって ' <1 ', ..., ' <55 ' とする。また, 3.3 節において述べた導出部分カーネルを用いた順序関数は 'dsk' とする。'dsk' のラベル付けの閾値には 5 秒を用いた。2,302 個の論理式から特徴ベクトルを抽出した後, 各特徴量毎に値が 0 から 1 の範囲に収まるように正規化した。

比較対象としてランダムな部分論理式の順序を用いて実験を行った。この実験は乱数の種を変更して 14 回繰り返し, 平均の計算時間及び各問題毎に最善・最悪の実行時間を記録した。

用意した順序関数を用いて 2,306 個の論理式に対し, QE 計算を実行した。表 5 に結果の統計を示す。"solved", "error", "timeout" の列はそれぞれ, QE 計算が 600 秒で終了した問題数, エラーで終了した問題数, 600 秒で終了しなかった問題数を表す。エラーは Maple 及び SyNRAC のバグ, またはメモリ不足のために生じた。"average", "average(log)", "median" の列はそれぞれ, QE 計算が 600 秒で終了した問題の平

表 2: N 秒以内に QE 計算が停止した論理式の数. これらの実行時間は ‘ADG’ の順序関数を用いて測定された.

N	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
#	433	846	1055	1261	1379	1476	1627	1694	1725
N	1.0	2.0	3.0	4.0	5.0	6.0	7.0	8.0	9.0
#	1752	1884	1937	1958	1977	1990	1999	2005	2013
N	10	20	30	100	150	200	300	500	600
#	2015	2040	2057	2096	2100	2102	2109	2113	2116

表 3: QE の計算時間と特徴ベクトルの例

Feature no.	time(sec)	1	2	3	4	5	6	7	8	9	10	11	12	13	
φ_1	0.078	1	1	1	1	1	2	1	1	1	2	1	0	0	
φ_2	0.037	1	1	1	1	1	1	1	1	1	1	1	0	0	
φ_3	1551.989	5	8	136	17	5	6.5	25	6	3	8	0	0	0	
Feature no.	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
φ_1	1	0	0	1	1	1	1	1	1	1	2	1	1	1	2
φ_2	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1
φ_3	0	0	0	1	1	4	8	125	15.6	4.6	6.5	21	5	3	8
Feature no.	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43
φ_1	1	0	0	1	0	0	1	1	0	0	0	0	0	0	0
φ_2	1	0	0	1	0	0	1	1	0	0	0	0	0	0	0
φ_3	0	0	0	0	0	0	1	1	1	3	11	3.7	1	7	5
Feature no.	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58
φ_1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
φ_2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
φ_3	1	1	8	3	0	0	1	0	0	1	1	1	0	0	0
Feature no.	x_{0-1}	x_{0-2}	x_{0-3}	x_{0-4}	x_{0-5}	x_{0-6}	x_{1-1}	x_{1-2}	x_{1-3}	x_{1-4}	x_{1-5}	x_{1-6}	x_{2-1}	x_{2-2}	x_{2-3}
φ_1	1	1	1	1	0	0	-	-	-	-	-	-	-	-	-
φ_2	-	-	-	-	-	-	1	1	1	1	0	0	-	-	-
φ_3	-	-	-	-	-	-	-	-	-	-	-	-	1	3	6
Feature no.	x_{2-4}	x_{2-5}	x_{2-6}	x_{3-1}	x_{3-2}	x_{3-3}	x_{3-4}	x_{3-5}	x_{3-6}	x_{4-1}	x_{4-2}	x_{4-3}	x_{4-4}	x_{4-5}	x_{4-6}
φ_1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
φ_2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
φ_3	8	8	0	1	1	5	3	3	0	1	1	6	7	7	0
Feature no.	x_{5-1}	x_{5-2}	x_{5-3}	x_{5-4}	x_{5-5}	x_{5-6}	x_{6-1}	x_{6-2}	x_{6-3}	x_{6-4}	x_{6-5}	x_{6-6}			
φ_1	-	-	-	-	-	-	-	-	-	-	-	-			
φ_2	-	-	-	-	-	-	-	-	-	-	-	-			
φ_3	1	1	6	3	3	0	0	1	6	3	3	0			

表 4: Decision Value の例

	φ_1	φ_2	φ_3	Order
<1	-4.143236	-4.126463	-7.026860	$\varphi_2 \rightarrow \varphi_1 \rightarrow \varphi_3$
<2	4.924164	4.659644	6.950127	$\varphi_3 \rightarrow \varphi_1 \rightarrow \varphi_2$
<3	2.092471	2.145601	4.441247	$\varphi_3 \rightarrow \varphi_2 \rightarrow \varphi_1$
<4	1.428065	1.430775	-0.009359	$\varphi_2 \rightarrow \varphi_1 \rightarrow \varphi_3$
<5	1.981705	2.103514	0.978644	$\varphi_2 \rightarrow \varphi_1 \rightarrow \varphi_3$
<6	2.063262	2.127833	-2.001929	$\varphi_2 \rightarrow \varphi_1 \rightarrow \varphi_3$
<7	1.462456	1.469122	3.834284	$\varphi_3 \rightarrow \varphi_2 \rightarrow \varphi_1$
<8	4.010731	4.015434	0.297530	$\varphi_2 \rightarrow \varphi_1 \rightarrow \varphi_3$
<9	25.655055	25.840633	-1.913830	$\varphi_2 \rightarrow \varphi_1 \rightarrow \varphi_3$
<10	36.241697	36.340194	-3.770675	$\varphi_2 \rightarrow \varphi_1 \rightarrow \varphi_3$
<20	38.722068	38.077265	6.001291	$\varphi_1 \rightarrow \varphi_2 \rightarrow \varphi_3$
<30	6.523585	6.529640	0.220255	$\varphi_2 \rightarrow \varphi_1 \rightarrow \varphi_3$
<55	21.408608	21.551922	-1.358177	$\varphi_2 \rightarrow \varphi_1 \rightarrow \varphi_3$
dsk	4841.204089	4841.204089	8.713206	$\varphi_1 \rightarrow \varphi_2 \rightarrow \varphi_3$

均実行時間, 実行時間の対数の平均, タイムアウト時の実行時間を無限大として扱ったときの実行時間の中央値を表す. “average” 及び “median” の単位は秒である. ‘rand_aver’ の行はランダムな順序関数を用いた実行時間の 14 回分の平均であり, ‘rand_best’ (‘rand_worst’) は各問題に関して 14 回の実行で最善 (最悪) の結果を集めた統計である.

定められた時間内に計算が停止するという性質が実用上重要であると考え, ここでは主に 600 秒以下で QE 計算が停止した問題の数で順序関数を評価することとする. このとき, ‘nvar_rev’, ‘npoly_rev’, ‘random’, ‘<1’, ‘<2’ はその他の順序関数に比べて効率が悪いといえる. ‘<1’ 及び ‘<2’ の効率が悪い原因は負例の中にも比較的簡単な問題が混ざっていたため, タイムアウトになるような本当に難しい問題を過小評価したからだと推定される.

順序関数 ‘dsk’ は ‘ADG’ や ‘<9’ に比べると効率が悪かった. RBF カーネルを使った機械学習に比べると図 2 に示したように手順が煩雑になるため実験が動作確認の段階に留まり, まだ調整の余地が残っていると考えられる. 論理式の特徴をよく捉えるカーネルの開発は今後の研究課題とする.

表 6 に 4 つのヒューリスティクス ‘ADG’, ‘nvar’, ‘npoly’, ‘sotd’ と 6 つの機械学習 ‘<1’, ‘<3’, ‘<5’, ‘<7’, ‘<9’, ‘dsk’ を用いた QE 計算時間の比較を示す. ‘id’ の列は論理式の識別番号を表し, 各行が対応する論理式の QE 計算にかかる時間を表す. “total” の行は表 6 に示した問題の内, 600 秒以内に QE 計算が終了した問題数を表す. 4.2 節で例として挙げた論理式 (1) は表 6 の識別番号 2,051 の論理式に当たる. 表 6 に挙げられていない論理式は順序関数の選択によらず 600 秒以内に QE が終了, またはタイムアウトとなった.

2,306 問の内, 順序関数の選択によって “solved” と “timeout” の結果が入れ替わったものは 40 問に満たなかった. 順序関数の選択による結果の変化が少ない問題にのみ表れることを考えれば, 改善幅が小さいものの, ‘<9’ などの機械学習により ‘ADG’ のようなヒューリスティクスと同等以上の順序関数を得られたことは重要である. 試行錯誤を繰り返して特徴量の最適な組み合わせを実現するヒューリスティクスを開発するには膨大な時間が必要である. 実際, 今回の実験では 1 回の実験につき 2,306 問に対する QE 計算を行うために 40 時間程度を要した. 効果的な順序関数が機械学習によって得られたという事実は機械学習がヒューリスティクス開発の時間を短縮するために役立つ可能性を示唆している.

表 5: 順序関数を変化させた際の実行時間に関する統計

	solved	error	timeout	average(sec)	average(log)	median(sec)
ADG	2116	3	187	4.770	-1.038	0.337
nvar	2115	4	187	4.741	-0.993	0.358
npoly	2114	4	188	4.594	-1.134	0.267
sotd	2114	4	188	4.269	-1.217	0.234
msotd	2114	4	188	4.240	-1.212	0.233
mtdeg	2114	4	188	4.172	-1.189	0.262
mdeg	2115	3	188	4.005	-1.230	0.222
mterm	2114	4	188	4.982	-1.150	0.256
nvar_rev	2097	2	207	4.857	-1.129	0.260
npoly_rev	2098	2	206	4.560	-1.119	0.268
rand_aver	2104.93	3.64	197.43	4.764	-1.182	0.241
rand_best	2121	2	183	4.569	-1.389	0.190
rand_worst	2087	2	217	4.531	-0.890	0.356
<1	2099	4	203	5.141	-0.934	0.300
<2	2102	3	201	5.320	-0.995	0.332
<3	2115	3	188	4.673	-0.928	0.360
<4	2112	4	190	4.732	-0.959	0.384
<5	2117	3	186	5.153	-0.928	0.392
<6	2115	2	189	4.540	-1.155	0.261
<7	2115	2	189	4.508	-1.178	0.244
<8	2113	3	190	4.806	-1.105	0.267
<9	2118	2	186	4.750	-1.165	0.239
<10	2117	2	187	4.578	-0.956	0.362
<20	2116	3	187	4.451	-1.200	0.235
<30	2114	2	190	4.886	-1.173	0.245
<55	2114	3	189	4.808	-0.993	0.351
dsk	2112	3	191	4.803	-1.114	0.275

表 6: 順序関数毎の QE 計算時間の比較. 実行時間の単位は秒である. TO は計算時間が 600 秒以上かかったことを示す. ERR は計算がエラーにより終了したことを示す.

id	ADG	nvar	npoly	sotd	<1	<3	<5	<7	<9	dsk
2051	0.12	0.13	0.12	0.12	0.19	TO	0.16	TO	0.15	0.18
1142	0.57	0.61	0.64	0.60	TO	0.63	0.64	0.65	0.63	0.63
1944	TO	TO	10.69	16.75	TO	6.92	8.24	17.40	4.61	16.13
889	5.01	5.74	5.52	5.56	TO	5.64	5.62	6.00	5.98	8.95
272	2.98	3.55	3.26	3.26	ERR	5.32	5.74	6.21	6.76	4.53
1024	8.15	11.87	10.20	10.22	TO	10.31	10.28	10.54	10.88	10.21
993	TO	9.74	TO	TO	TO	10.44	10.05	10.92	13.45	TO
999	TO	TO	TO	TO	12.54	13.33	13.35	13.30	13.57	TO
1528	22.28	25.46	24.57	23.90	TO	25.06	23.27	23.60	23.24	25.49
2167	28.49	114.20	21.46	21.31	TO	48.71	27.14	24.27	27.32	30.86
2294	37.91	TO	31.60	31.80	TO	48.95	28.88	27.41	31.93	TO
1088	58.49	TO	TO	TO	TO	74.81	32.61	TO	32.43	261.20
1039	31.24	32.11	32.38	31.65	32.27	32.46	32.44	33.46	32.69	34.92
1112	40.31	39.73	39.03	40.14	120.55	40.05	39.67	TO	39.40	39.86
1234	45.87	49.48	41.81	42.72	14.92	39.83	39.78	15.17	44.14	48.31
32	42.21	43.80	292.48	44.66	TO	41.92	44.04	43.76	44.25	TO
1239	39.87	48.73	43.35	44.19	13.44	44.93	40.2	13.79	44.94	48.56
2045	67.18	60.06	61.08	60.17	TO	71.15	TO	60.30	TO	TO
1101	61.42	61.16	63.67	61.85	TO	62.39	61.29	60.74	62.39	61.79
2168	62.42	62.72	64.16	61.87	TO	60.44	153.39	62.74	74.04	69.02
33	69.83	75.44	76.99	76.54	TO	69.87	68.68	76.28	75.79	81.72
1040	80.14	87.35	107.34	87.68	TO	TO	89.18	89.93	87.61	107.17
28	199.02	208.30	TO	TO	TO	TO	202.83	210.17	209.45	TO
151	239.64	218.08	244.91	246.53	TO	243.18	236.42	243.94	248.93	259.85
1125	TO	TO	TO	TO	TO	252.36	252.79	296.94	250.25	TO
1092	266.09	153.79	TO	149.94	323.97	TO	TO	TO	275.69	175.30
5	279.42	285.78	295.58	289.15	TO	283.14	277.4	291.48	292.78	307.87
17	341.99	293.81	230.59	345.67	272.41	352.92	307.32	242.58	339.28	324.76
112	TO	TO	TO	TO	409.60	TO	TO	TO	TO	TO
610	437.12	455.59	456.30	TO	393.91	TO	TO	TO	TO	469.32
1091	523.86	528.97	476.57	466.62	TO	475.28	521.76	363.77	472.13	516.87
2216	525.31	482.94	485.39	481.33	TO	573.73	566.82	482.50	487.02	514.58
150	513.11	592.43	589.52	596.24	586.34	571.58	548.01	586.74	586.56	TO
total	28	27	26	26	11	27	29	27	30	24

5 結論

今回の実験により部分論理式の順序が非冠頭標準形の論理式に対する QE 計算の効率に与える影響が検証された。ヒューリスティクスによる順序付けはランダムな順序に比べて効率が高かったが、訓練データのラベル付けが適切になされた場合の機械学習を用いた順序付けも同等以上に効率が高かった。この結果は機械学習の応用がヒューリスティクス開発の試行錯誤に費やす手間を大幅に省く可能性を示唆している。また、論理式専用にカーネルを設計するという方針に関して、本研究の現状では動作確認に留まった。論理式を評価するために最適化されたカーネル設計の研究は今後の課題である。

論理式中に表れる有用なパターンを効果的に利用するために数式処理へ機械学習を応用する更なる研究が必要である。本研究は数式処理のアルゴリズムにおいてもヒューリスティクス選択の場面で機械学習が有用な選択肢となりうることを示唆している。

謝辞

本研究を進めるに当たり、富士通研究所 穴井宏和 氏と名古屋大学 松崎拓也 准教授に研究課題の設定及び文書執筆時の校正作業に関し多大なご助力をいただきました。また、プロジェクト「ロボットは東大に入れるか」から資金援助をいただきました。これらのご支援にこの場を借りてお礼申し上げます。

参 考 文 献

- [1] Noriko H. Arai, Takuya Matsuzaki, Hidenao Iwane, and Hirokazu Anai. Mathematics by machine. In *Proceedings of the 39th International Symposium on Symbolic and Algebraic Computation*, pp. 1–8. ACM, 2014.
- [2] Christopher W Brown. Simple cad construction and its applications. *Journal of Symbolic Computation*, Vol. 31, No. 5, pp. 521–547, 2001.
- [3] Christopher W. Brown and James H. Davenport. The complexity of quantifier elimination and cylindrical algebraic decomposition. In *Proceedings of the 2007 International Symposium on Symbolic and Algebraic Computation*, ISSAC '07, pp. 54–60, New York, NY, USA, 2007. ACM.
- [4] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, Vol. 2, No. 3, p. 27, 2011.
- [5] Michael Collins and Nigel Duffy. Convolution kernels for natural language. In *Advances in neural information processing systems*, pp. 625–632, 2001.
- [6] Andreas Dolzmann, Andreas Seidl, and Thomas Sturm. Efficient projection orders for CAD. In *Proceedings of the 2004 international symposium on Symbolic and algebraic computation*, pp. 111–118. ACM, 2004.
- [7] David Haussler. Convolution kernels on discrete structures. Technical report, Citeseer, 1999.
- [8] Zongyan Huang, Matthew England, David Wilson, James H. Davenport, Lawrence C. Paulson, and James Bridge. Applying machine learning to the problem of choosing a heuristic to select the variable ordering for cylindrical algebraic decomposition. In *Intelligent Computer Mathematics*, pp. 92–107. Springer, 2014.

- [9] Hidenao Iwane, Hiroyuki Higuchi, and Hirokazu Anai. An effective implementation of a special quantifier elimination for a sign definite condition by logical formula simplification. In *Computer Algebra in Scientific Computing*, pp. 194–208. Springer, 2013.
- [10] Hidenao Iwane, Takuya Matsuzaki, Noriko H. Arai, and Hirokazu Anai. Automated natural language geometry math problem solving by real quantifier elimination. In *Proceedings of the 10th International Workshop on Automated Deduction in Geometry*, pp. 75–84, 2014.
- [11] Hidenao Iwane, Hitoshi Yanami, and Hirokazu Anai. SyNRAC: A toolbox for solving real algebraic constraints. In *Mathematical Software-ICMS 2014*, pp. 518–522. Springer, 2014.
- [12] Takuya Matsuzaki, Hidenao Iwane, Hirokazu Anai, and Noriko H. Arai. The most uncreative examinee: a first step toward wide coverage natural language math problem solving. In *Twenty-Eighth AAAI Conference on Artificial Intelligence*, pp. 1098–1104, 2014.
- [13] John Shawe-Taylor and Nello Cristianini. *Kernel methods for pattern analysis*. Cambridge university press, 2004.
- [14] 鹿島久嗣. カーネル法による構造データの解析 (機械学習による自然言語処理・言語処理を利用したメディア理解, 学習理論とパターン認識メディア理解, 機械学習による自然言語処理・言語処理を利用したメディア理解, 一般). 電子情報通信学会技術研究報告. PRMU, パターン認識・メディア理解, Vol. 104, No. 670, pp. 61–66, 2005.
- [15] 鹿島久嗣, 坂本比呂志, 小柳光生. 木構造データに対するカーネル関数の設計と解析. 人工知能学会論文誌, Vol. 21, pp. 113–121, 2006.