ソルバを利用した弱適切なポリシの導出方法の提案と評価

2020SC008 二村公彗　2020SC090 田中美里
指導教員：石原靖哲

1　はじめに

近年，企業ではあらゆる情報が電子データ化し，データ交換の技術が必要となってきている．しかし，データを交換する際には個人情報を外部に漏らさないために，プライバシやセキュリティを確保する必要がある．そこで，本研究では，ソルバを利用してセキュリティを確保する方法の提案と評価を行う．

1.1　問題設定

[1] では，一次情報提供者，二次情報提供者，そしてそれらが公開したデータを受け取るデータユーザの三種類の参加者がデータのやり取りを行う状況を想定している．そこでは，一次情報提供者がソースデータを，二次情報提供者ターゲットデータを所持し，それらのデータをデータユーザに公開する (図 1)．一般に，一次情報提供者はデータユーザに対するデータ公開ポリシ Q_S を持ち，二次情報提供者もまた，ポリシ Q_T を持つ．相崎らの研究 [1] では，**弱適切なポリシ**であるための条件として次の二つを定義している．

1. Q_T は秘匿性を満たす．すなわち，$Q_T \circ \mathbf{M}$ の答えは Q_S の答えを用いて必ず求めることができることを指す．ここで，$\circ$ は問合せの合成を表す．
2. Q_T は何らかの情報を公開する．

例 1　X 市の学校で働く教員に関するデータを収集する X 市教育委員会と，X 市教育委員会から提供されたある公立校 a の教員に関するデータを管理する公立校 a が存在する状況を考える．X 市教育委員会はその一部のデータを X 市教育委員会の公式ホームページに公開し，公立校 a はその一部のデータを公立校 a の公式ホームページに公開する．この例において，X 市教育委員会は一次情報提供者，公立校 a は二次情報提供者，公式ホームページの閲覧者はデータユーザに当たる．X 市教育委員会は関係スキーマ $S(\mathrm{Name}, \mathrm{Age}, \mathrm{Gender}, \mathrm{Work})$ 上のデータを保持しているとする．これは，S という名前のテーブルに，氏名，年齢，性別，勤務校のデータを保持していることを表す．このとき，X 市教育委員会と公立校 a の間のデータ交換メカニズムは次の連言問合せで表現される．

$$\mathbf{M}: \quad T(Nam, Age, Gen) \text{ :- } S(Nam, Age, Gen, \text{“a”}).$$

これは，テーブル S から勤務校が a であるデータを全て取り出し，テーブル T に氏名，年齢，性別を格納することを表す．X 市教育委員会は，公式ホームページを作成する際に個人情報である氏名の情報を秘匿したいと考え，Q_S として次のような問合せを与えたとする．

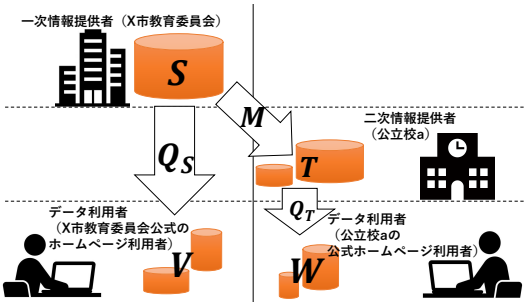

図 1　問題設定

$$Q_S: \quad V(Age, Gen, Wor) \text{ :- } S(Nam, Age, Gen, Wor).$$

ここで，公立校 a がデータ公開ポリシとして，次のような連言問合せ Q_T を採用した場合を考える．

$$Q_T: \quad W(Age, Gen) \text{ :- } T(Nam, Age, Gen).$$

このとき，$Q_T \circ \mathbf{M}$ の答えは Q_S の答えを用いて必ず求めることができるため秘匿性を満たし，かつ何らかの情報を公開するので，Q_T は「弱適切なポリシである」という．

一方で，X 市教育委員会が教員の氏名と勤務校の情報を秘匿したい場合の連言問合せ Q'_S を考える．

$$Q'_S: \quad V(Age, Gen) \text{ :- } S(Nam, Age, Gen, Wor).$$

そして，公立校 a のデータ公開ポリシとして，上と同じ Q_T を採用した場合を考える．一見すると，Q_T が公開する情報は Q'_S が公開する年齢と性別のデータと同じであり秘匿性を満たすように考えられるが，公立校 a は勤務校が公立校 a に関するデータのみを得ているため，X 市教育委員会が得ることのできない，教員の勤務校の情報を一部得ていることになる．実際，Q_S の答えを用いて $Q_T \circ \mathbf{M}$ の答えを求められない場合が存在するため秘匿性を満たしていない．なお，この後半の例において秘匿性を満たす Q_T は何も情報を公開しないポリシのみであり，弱適切なポリシは存在しないことが確認できる．■

1.2　先行研究の課題と本研究の成果

我々の先行研究 [1] では，与えられた Q_S と $\mathbf{M}$ に対して，弱適切なポリシという概念を導入した．さらに，弱適切なポリシが存在するための十分条件を判定するアルゴリズムを提案し，その判定が必要条件の判定にもなるための制約を与えた．[1] で提案されたアルゴリズムとは，Q_S を使った問合せ N を総当たりで発生させ，$N \circ Q_S$ と $Q_T^N \circ \mathbf{M}$ が等価となるような Q_T^N が存在するかを判定するアルゴリズムである．

しかし，[1] では Q_S を使った問合せ N を総当たりで発生させることから，$N \circ Q_S$ の総数が増加し，$N \circ Q_S$ と $Q_T^N \circ \mathbf{M}$ の等価性判定に莫大な処理時間を要する恐れがある．そこで，[2] では，実用の場面で役に立つのかを確認するために，[1] で提案されたアルゴリズムの実装と性能評価を行った．実装には Python と Prolog を用い，実用的な時間内に動作するには Q_S の頭部の引数が 8 個程度までであるということがわかった．ただし，M の問合せが 2 つの場合に限られている点や $N \circ Q_S$ の総数の上限など，実装上の課題が残されている．

そこで我々は，[1] で提案されたアルゴリズムの総当たりの発生をソルバに解かせることで弱適切なポリシを導く方法の提案と評価を行う．与えられた Q_S と $\mathbf{M}$ に対して，$R \circ Q_S$ と $Q_T \circ \mathbf{M}$ が等価となるような R と Q_T が存在するための制約式を生成してソルバに解かせることで，弱適切なポリシ Q_T を導出する．本研究では，Python に Z3 ソルバを導入して実装を行う．Z3 ソルバとは，Microsoft 社が開発した SMT ソルバである [3]．これは整数や実数上の不等式を含む充足問題にも適用でき特に最適化問題に特化している．そして，性能評価を行うため [2] で課題となったプログラムのターンアラウンド時間について，[2] との比較を行った．実行した結果 [2] よりもターンアラウンド時間が短くなった．

2 諸定義

この節では [1]，[2] に基づいて諸定義を与える．

2.1 関係データベース

関係スキーマ $R[n]$ は関係データの構造を表すものであり，形式的には関係名 R とアリティと呼ばれる非負整数 n から成る．関係スキーマ上の実データは**タプル** t の集合で与えられる．$R[n]$ 上のタプルは n 個のエントリをもつ．各エントリは，**ドメイン**と呼ばれる，定数の可算集合 Dom の要素をとる．**関係インスタンス** I はタプルの有限集合である．

データベーススキーマ $\mathbf{R}$ は相異なる関係名をもつ関係スキーマの有限リスト $\mathbf{R} = \langle R_1[n_1], \ldots, R_k[n_k] \rangle$ であり，関係データベースの構造を表す．**データベースインスタンス** I はリスト $\langle I_1, \ldots, I_k \rangle$ である．ただし，各 I_i は $R_i[n_i]$ の関係インスタンスである．$\mathbf{I}$ の i 番目の関係インスタンスがタプル $(a_1, \ldots, a_{n_i})$ を含むとき，$R_i(a_1, \ldots, a_{n_i}) \in \mathbf{I}$ と記す．

$\mathbf{R} = \langle R_1[n_1], \ldots, R_k[n_k] \rangle$ をデータベーススキーマとする．変数の可算集合 Var を固定する．$\mathbf{R}$ における**原子式**は $R_i(x_1, \ldots, x_{n_i})$ と表される．ただし，各 x_i は Dom ∪ Var に属する．$\mathbf{R}$ における**連言式**は原子式のリストであり，空の場合 $\top$ と記す．$\mathbf{R}$ における**連言問合せ**（以降，CQ と記す）Q は次の形式で与えられる規則である:

$$Q: \quad V(y_1, \ldots, y_m) \text{ :- } R_1, \ldots, R_\ell.$$

ただし，V はアリティ m の，$\mathbf{R}$ に属さない関係名であり，各 y_i は Dom ∪ Var に属する．$R_1, \ldots, R_\ell$ は $\mathbf{R}$ における連言式である．$V(y_1, \ldots, y_m)$ を Q の**頭部**，$R_1, \ldots, R_\ell$ を Q の**体部**と呼ぶ．すべての $y_1, \ldots, y_m$ が相異なる変数であり，かつ体部にも現れるとき，Q は**安全**であるという．

$\mu : \text{Var} \to \text{Dom}$ を変数割当てとする．$\mathbf{R}$ 上の $\mathbf{I}$ における Q の **答え** $Q(\mathbf{I})$ は次のように表される．

$$Q(\mathbf{I}) = \{(\mu(y_1), \ldots, \mu(y_m)) \mid \text{各 } R_i(x_1, \ldots, x_{n_i}) \text{ に対し } R_i(\mu(x_1), \ldots, \mu(x_{n_i})) \in \mathbf{I}\}$$

Q が安全であるならば $Q(\mathbf{I})$ は有限であり，したがって $Q(\mathbf{I})$ は $V[m]$ 上の関係インスタンスである．

安全な CQ Q の体部に現れる変数のうち，頭部にも現れる変数を *distinguished* な変数と呼び，頭部には現れない変数を *non-distinguished* な変数と呼ぶ．CQ Q の**代入** θ は，Q に現れる distinguished な変数を Var ∪ Dom の要素に写す写像である．

2.2 問合せの等価性と書き換え

$\mathcal{I}(\mathbf{R})$ を $\mathbf{R}$ 上のすべてのデータベースインスタンスの集合とし，Q_1 と Q_2 を $\mathbf{R}$ 上の CQ とする．任意の $\mathbf{I} \in \mathcal{I}(\mathbf{R})$ に対し $Q_1(\mathbf{I}) \subseteq Q_2(\mathbf{I})$ であるとき，Q_1 は Q_2 に含まれるといい，$Q_1 \subseteq Q_2$ と記す．$Q_1 \subseteq Q_2$ であることと Q_2 から Q_1 への準同型写像が存在することは同値である．$Q_1 \subseteq Q_2$ かつ $Q_2 \subseteq Q_1$ であるとき，Q_1 と Q_2 は**等価**であるといい，$Q_1 \equiv Q_2$ と記す．

information-collapsing な CQ とは，$Q_\perp : W() \text{ :- } \top$ の形式で与えられる nullary な（アリティが 0 の）CQ である．これは，どんなデータベースインスタンスに対しても空タプル（エントリ数が 0 のタプル）のみからなる集合を答えとして返すため，何も情報を公開しないポリシに相当する．安全な CQ Q について，$Q \not\equiv Q_\perp$ であるとき，Q は *information-revealing* であるという．これは，何らかの情報を公開するポリシに相当する．

$\mathbf{M}$ を安全な CQ のリストとする．$\mathbf{M}$ を**使った** CQ R とは，R の体部に現れるのは $\mathbf{M}$ の頭部に現れる関係名のみであるような CQ である．安全な CQ Q に対し，$\mathbf{M}$ を使った Q の *CQ-rewriting* とは次のような CQ R である (図 2)．

- R は $\mathbf{M}$ を使った CQ である．
- $R \circ \mathbf{M} \equiv Q$ が成り立つ．ただし，$\circ$ は問合せの合成を表す．

2.3 弱適切なターゲットポリシ

安全な CQ Q_S と安全な CQ のリスト $\mathbf{M}$ が与えられているとする．Q_S を使った $Q_T \circ \mathbf{M}$ の CQ-rewriting が存在するとき Q_T は秘匿性を満たすという．Q_T が秘匿性を満たし，かつ information-revealing であるとき，Q_T は Q_S と $\mathbf{M}$ に関して**弱適切なポリシ**であるという．

例 2 例 1 と同じ CQ リスト $\mathbf{M}$ と CQ Q_S, Q_T を考える．

$$\begin{aligned}\mathbf{M}:&\quad T(Nam, Age, Gen) \text{ :- } S(Nam, Age, Gen, \text{“a”}).\\ Q_S:&\quad V(Age', Gen', Wor')\\ &\qquad\qquad \text{:- } S(Nam', Age', Gen', Wor').\\ Q_T:&\quad W(Age, Gen) \text{ :- } T(Nam, Age, Gen).\end{aligned}$$

Q_S を使った次の CQ R を考える．

$$R:\quad W(Age', Gen') \text{ :- } V(Age', Gen', Wor').$$

次の CQ $Q_T \circ \mathbf{M}$ と $R \circ Q_S$ を得る．

$$\begin{aligned}Q_T \circ \mathbf{M}:&\quad W(Age, Gen) \text{ :- } S(Nam, Age, Gen, \text{“a”}).\\ R \circ Q_S:&\quad W(Age', Gen') \text{ :- } S(Nam', Age', Gen', \text{“a”}).\end{aligned}$$

このとき，$Q_T \circ \mathbf{M}$ から $R \circ Q_S$ への準同型写像 $h = \{Nam \mapsto Nam', Age \mapsto Age', Gen \mapsto Gen'\}$ が存在し，かつ $R \circ Q_S$ から $Q_T \circ \mathbf{M}$ への準同型写像 $g = \{Nam' \mapsto Nam, Age' \mapsto Age, Gen' \mapsto Gen\}$ が存在するため，$R \circ Q_S \equiv Q_T \circ \mathbf{M}$ が成立する．よって，Q_T は Q_S と $\mathbf{M}$ に関して弱適切なポリシである． ■

3 弱適切なポリシの導出方法

与えられた Q_S を用いた CQ R と，与えられた $\mathbf{M}$ を用いた CQ Q_T について，$R \circ Q_S$ と $Q_T \circ \mathbf{M}$ を導き出す必要がある．その上で，$R \circ Q_S \equiv Q_T \circ \mathbf{M}$ が成り立つような R と Q_T が存在すれば，そのときの Q_T は「弱適切なポリシである」といえる．そこで本研究では，以下の対応づけのもと，$R \circ Q_S \equiv Q_T \circ \mathbf{M}$ に相当する Z3 の制約式を生成し，それを解かせることで弱適切なポリシを導出する．

- 問合せの変数や定数を，Z3 の定数に対応させる
- $R \circ Q_S$ や $Q_T \circ \mathbf{M}$ などの問合せの合成を表現するために必要な問合せへの代入を，Z3 の定数から定数への関数に対応させる
- 問合せ間の準同型写像を，Z3 の定数から定数への関数に対応させる

例 2 をもとに説明を行う．まず, 与えられた問合せ $\mathbf{M}$ の変数である Nam, Age, Gen と定数である a と, Q_S の変数である Nam', Age', Gen', Wor' を Z3 の定数に対応させるための宣言が必要となる．そして，$\mathbf{M}$ の distinguished な変数 Nam，Age，Gen と，Q_S の distinguished な変数 Age'，Gen'，Wor' を要素に写す代入を，Z3 の定数から定数への関数に対応させるための宣言が必要となる．最後に，$R \circ Q_S$ と $Q_T \circ \mathbf{M}$ の間の準同型写像を，Z3 の定数から定数への関数に対応させるための宣言が必要となる．これらの方針に沿って， 4 節では詳細なプログラムの処理や関数についての説明を行う．

4 実装と評価

4.1 実装環境

本研究の実装環境は以下の通りである．

- プログラミング言語：Python 3.10.9
- 開発環境：Spyder 5.4.3
- SMT ソルバ：Z3-solver 4.12.2.0

4.2 プログラム内容

3 節の導出方法を実装するにあたり，本研究で作成したプログラムを約 240 行で記述している．弱適切なポリシ Q_T を求めるため，問合せ $\mathbf{M}$ と Q_S を Python の文字列として与え，それぞれ Z3 に対応させる．その後，問合せへの代入や問合せ間の準同型写像を意味する Z3 の関数を作成し，各関数が存在するための制約を与え，その制約を満たす解を Z3 に見つけさせる．この一連の流れを本研究で作成したプログラムは行っている．

以下の 4.2.1，4.2.2，4.2.3 節では，プログラムの中で用いた関数や具体的な処理について記述する．

4.2.1 問合せの変数や定数を Z3 の定数に対応させる処理

入力として Python の文字列で記述された問合せを与えるが，Python で記述した状態では Z3 に計算させることができないため，Python の文字列を Z3 の文字列に対応させる．そのために作成したのが *parse* **関数**と *declareObj* **関数**である．parse 関数は，Python の文字列として与えられた問合せを構文解析する関数である．declareObj 関数は，Python の文字列リストを読み込み，各文字列を Z3 の文字列に対応させる関数である．

4.2.2 問合せへの代入に対応する Z3 の関数

問合せ $\mathbf{M}$ や Q_S への代入を表す，Z3 の関数 *thetaMM* **関数，***thetaQs* **関数**を用意する．Z3 の関数とは Z3 によって探索される解の一部である．それらの関数に対して，以下の制約を生成する．

- thetaMM 関数と thetaQs 関数は定数を変化させない
- thetaMM 関数と thetaQs 関数は non-distinguished な変数を変化させない
- thetaMM 関数と thetaQs 関数は distinguished な変数を non-distinguished な変数に一致させられない

これによって，Z3 に問合せへの代入を見つけさせる．

4.2.3 問合せ間の準同型写像に対応する Z3 の関数

Z3 に問合せ間の準同型写像を見つけさせるための，Z3 の関数 *hMMtoQs* **関数，***hQstoMM* **関数**を用意する．この関数はそれぞれ，$Q_T \circ \mathbf{M}$ から $R \circ Q_S$ への準同型写像，$R \circ Q_S$ から $Q_T \circ \mathbf{M}$ への準同型写像を格納している．これらの関数に対して以下の制約を生成する．

- hMMtoQs 関数と hQstoMM 関数は定数を変化させない
- $Q_T \circ \mathbf{M}$ の体部の各原子式に hMMtoQs 関数を適用した式が $R \circ Q_S$ の体部に現れる

- $R \circ Q_S$ の体部の各原子式に hQstoMM 関数を適用した式が $Q_T \circ \mathbf{M}$ の体部に現れる

これにより，Z3 にそれぞれの問合せ間の準同型写像を見つけさせる．

4.3 実行結果

本研究のプログラムを以下の問合せに対して実行した．

$$\mathbf{M}: \quad t_1(Y_1) \text{ :- } s(Y_1, Y_2), \quad t_2(Z_1) \text{ :- } s(Z_1, Z_2).$$

$$Q_S: \quad v(X_1, X_2) \text{ :- } s(X_1, X_2).$$

関係名である t_1 や s に対応する Z3 の文字列定数 t1 や s，non-distinguished な変数である Y_2 に対応する Z3 の文字列定数 Y2@MM0_0 等は変化しておらず，それ以外の変数に対しては Z3 が各変数に代入される値を見つけてきていることが確認できた．また，違う問合せを用いても同様に Z3 が正しい代入や準同型写像を見つけてくること，さらに，弱適切なポリシが存在しない場合の問合せを用いた時には，弱適切なポリシが存在しないことを表す「unsat」が正しく出力されることを確認した．よって，このプログラムは意図した通りに動作しており，弱適切なポリシを導出することが出来たと言える．

5 性能評価

作成したプログラムの評価方法と評価結果について述べる．性能評価をするにあたり用いたプログラムは，[2] で実装したプログラム (以降 okaPGM と記す)，本研究で実装した，問合せを Z3 の文字列に対応させたプログラム (以降 strPGM と記す)，strPGM を元に問合せを Z3 の整数に対応させたプログラム (以降 intPGM と記す) である．また，性能評価に用いた PC のスペックは以下の通りである．

- CPU：Intel(R) Core i5-8250U 1.60GHz
- RAM：16.0 GB
- OS：Windows 11 home ver22H2

5.1 評価方法

4.3 節の問合せが与えられ，問合せ Q_S の引数と $\mathbf{M}$ の体部の変数を 1 個ずつ増やしていった場合のプログラムのターンアラウンド時間と使用メモリ量について，time モジュールと tracemalloc というライブラリを用いて，各引数に対して 10 回ずつプログラムを実行させたその平均を取ることで性能評価を行う．

5.2 評価結果

ターンアラウンド時間，使用メモリ量で比較した結果が表 1，表 2 である．

okaPGM と比べると，strPGM，intPGM の方がターンアラウンド時間が短くなっており，okaPGM では求められなかった引数 6 個以上の問合せでも弱適切なポリシを導出できることが分かった．また，使用メモリ量は strPGM よりも，intPGM の方が少ない結果となった．

表 1 ターンアラウンド時間 [s] の比較

引数の数 [個]	okaPGM	strPGM	intPGM
2	2.576	3.168	1.125
3	8.253	13.748	1.166
4	60.519	41.519	1.831
5	694.087	130.750	2.013
6	測定不可能	443.873	2.688
7	測定不可能	1750.494	3.360

表 2 使用メモリ量 [MB] の比較

引数の数 [個]	strPGM	intPGM
2	11.504	9.049
3	16.216	9.594
4	20.523	9.909
5	24.998	9.930
6	29.510	10.206
7	33.585	10.223

6 まとめと今後の課題

本研究では，ソルバを用いた弱適切なポリシの導出方法の提案と評価を行った．与えられた Q_S と $\mathbf{M}$ に対して，$R \circ Q_S$ と $Q_T \circ \mathbf{M}$ が等価となるような R と Q_T が存在するための制約式を生成してソルバに解かせることで，弱適切なポリシ Q_T を導出する方法を提案し，プログラムを作成した．その結果，先行研究でなされた実装よりもターンアラウンド時間が短く引数が 6 個以上でも対応可能なこと，Z3 の文字列に対応させるより Z3 の整数に対応させた方がターンアラウンド時間は短く，使用メモリ量も少ないためより実用的であることが分かった．

しかし，Q_S の引数の数や $\mathbf{M}$ の体部の変数の数の変化には対応させることが出来たが，$\mathbf{M}$ や Q_S の問合せを $t_1, ..., t_m$, $v_1, ..., v_m$ と増加させた場合にも本研究のプログラムが正しく動作するのかについては検討できなかった．今後の課題として，複数の問合せを用いた場合にも弱適切なポリシを導くことが可能か検討することが挙げられる．

参考文献

[1] 相崎聖也, 石原靖哲. 関係データ交換フレームワークにおける弱適切なデータ公開ポリシの導出アルゴリズム. 電子情報通信学会論文誌 D, Vol. J106-D, No. 4, pp. 257–267, 2023.

[2] 石田潤, 岡田哲典. 関係データ交換フレームワークにおける弱適切なポリシの導出アルゴリズムの実装と性能評価. 2022 年度南山大学理工学部機械電子制御工学科卒業論文, 2023.

[3] Nikolaj Bjørner and Lev Nachmanson. Z3-microsoft research. https://www.microsoft.com/en-us/research/project/z3-3/.