オブジェクト指向言語の情報流解析における機密度のパラメータ化

Secrecy Parameterization in Object-oriented Programs for Information Flow Analysis

吉田 真也 * **桑原 寛明** † **國枝 義敏** ‡

Summary. In this paper, we introduce a parametric polymorphism about secrecy for information flow analysis to Banerjee's object-oriented programming language. Information flow analysis is useful to detect invalid information leaks. However, it is difficult to describe classes for any secrecy of data such as Collection Framework, since type-based information flow analysis requires specifying a secrecy of each data as a type in the program. We propose a way to declare classes parameterized over secrecy and a structure of secrecy as a type. We also propose typing rules to ensure that typable programs do not leak confidential information even if any secrecy is substituted for the secrecy parameter. We show an application of our typing rules to a simple collection class as an example.

1 はじめに

個人情報などの機密情報を扱うプログラムは，機密情報を外部に漏洩しないことが求められる．機密情報の流出の可能性をプログラムの静的解析によって検出する手法として，型検査に基づく情報流解析が提案されている [1] [2] [3] [4]．型検査に基づく情報流解析では，プログラム中のデータの機密度を型とみなし，プログラムが非干渉性を満たすことを検査する型システムを構築する．非干渉性とは機密度の低いデータが機密度の高いデータに依存しないことを保証する性質である．オブジェクト指向プログラム向けの型検査に基づく情報流解析は Banerjee らによって提案されている [2]．黒川らは文献 [2] を例外処理付きオブジェクト指向言語向けに拡張した [4]．これらの型システムはいずれも非干渉性に対して健全であることが証明されている．

文献 [2] [4] ではフィールドや変数の型として格納されるデータの機密度を指定する．この時，機密度を具体的に指定する必要があるため，扱うデータの機密度を事前に決定できない汎用的なコレクションフレームワークなどのプログラムを記述する場合，必要な機密度の組み合わせごとにプログラムを記述しなければならない．機密度の種類と大小関係を定義する機密度束は一意ではないため，機密度束が異なれば同じように必要な機密度の組み合わせごとにプログラムを記述する必要もある．これらのプログラムは変数の型やメソッドのシグネチャに指定される機密度が異なるだけで，その他の構造や論理は同じである．指定される機密度のみが異なるプログラムが多数存在すると，構造や論理を修正，改良するときはすべてのプログラムを変更しなければならず，保守性が低下する．この問題を解決するためには，プログラム内で指定される機密度をパラメータ化できればよい．

本稿では Banerjee らが提案するオブジェクト指向言語に対して機密度をパラメータに取るクラスを記述できるように構文と型付け規則を拡張し，機密度に関するパラメトリックな多相性を導入する．プログラム中で指定する機密度は機密度定数と機密度パラメータのリストの組で表される．機密度パラメータに機密度定数を割り

*Shinya Yoshida, 立命館大学 大学院 情報理工学研究科

†Hiroaki Kuwabara, 南山大学 情報センター

‡Yoshitoshi Kunieda, 立命館大学 情報理工学部

当てることで機密度は具体化される．具体化された機密度の値は機密度にもともと含まれる機密度定数と機密度パラメータに割り当てられた機密度定数すべての結びである．型付けできるクラスはいかなる機密度定数で機密度パラメータが具体化されても非干渉性を満たすように型付け規則を定める．提案する言語に基づいてコレクションクラスの実装と利用の例を示し，提案する型付け規則で不正な情報流の有無を適切に検査できることを示す．

本稿の構成は以下のとおりである．2 章で型検査に基づく情報流解析と機密度のパラメータ化の必要性を述べる．3 章で，オブジェクト指向言語に対してパラメータ化された機密度を導入し，型付け規則について述べる．4 章で簡単な適用例を示す．5 章で関連研究を挙げ，6 章でまとめる．

2 情報流解析における機密度

情報流解析では，データの機密度に基づき不正な情報流の有無を検査する．低い機密度のデータが高い機密度のデータに直接あるいは間接的に依存するときに不正な情報流が存在するとみなす．機密度束 $(\mathcal{H}, \sqsubset)$ が機密度の種類と大小関係を定義する [5]．機密度 $\eta_L, \eta_H \in \mathcal{H}$ について，$\eta_L \sqsubset \eta_H$ のとき η_L は η_H と同じかそれより低い機密度である．型検査に基づく情報流解析では，データの型として機密度を与え，プログラムが非干渉性を満たすことを検査する型システムを構築する．

文献 [2] [4] のオブジェクト指向言語に対する情報流解析では，適切な機密度束を定義した上で，プログラム中に変数などの型として機密度を指定することが要求される．そのため，コレクションフレームワークなど扱うデータの機密度を事前に決定できないプログラムを作成する時は，必要な機密度の組み合わせごとにクラスを定義する必要がある．例えば，機密度束が $(\{L, H\}, \sqsubset)$ である時に，2 項組のクラス `Pair` を定義する場合，機密度が L と L の 2 項組のクラス `PairLL`，L と H の 2 項組のクラス `PairLH`，H と L の 2 項組のクラス `PairHL`，H と H の 2 項組のクラス `PairHH` をそれぞれ定義する必要がある．こうしたクラス定義の数は対象とする機密度束の元の数やクラス内で扱われるデータの数が増えると指数関数的に増加する．文献 [2] [4] のオブジェクト指向言語で記述した `PairLL` と `PairLH` のクラス定義を図 1 に示す．文献 [2] [4] のオブジェクト指向言語では，データ型と機密度の組が変数の型として指定される．メソッド宣言ではメソッドのヒープエフェクトが仮引数のリストの後に指定される．ヒープエフェクトはインスタンス生成やフィールド代入によって変更されるヒープ上のデータの機密度である．それぞれのクラスには 2 項組の各要素が格納されるフィールド `first` と `second` とそれぞれのアクセサメソッドが定義される．変数 `result` に代入された値がメソッドの戻り値になる．`PairLL` では `first` と `second` の機密度が L として宣言されており，`PairLH` ではそれぞれ L と H として宣言されている．これらのクラス定義はクラス中に指定される機密度が異なるだけで実装は同じである．機密度を除く定義や実装が同じクラスが多数存在すると，定義や実装を修正，改良するときはすべてのクラスを変更しなければならず，保守性が低下する．そこで本稿では, この問題を解決するためにクラス定義における機密度のパラメータ化を導入し，機密度パラメータにどのような機密度が割り当てられても不正な情報流が存在しないことを保証できるような型付け規則を構築する．

3 機密度のパラメータ化

コレクションフレームワークなど具体的な機密度を後から指定可能なクラスを記述できるように，機密度をパラメータに取るクラス定義を導入する．プログラム中の機密度パラメータにいかなる機密度が指定されても型付け可能なプログラムには不正な情報流が存在しないように型付け規則を定める．以下では，機密度束 $(\mathcal{H}, \sqsubset)$ を仮定し，$\eta \in \mathcal{H}$ を機密度定数と呼ぶ．$\bot_{\mathcal{H}}$ は機密度束 $(\mathcal{H}, \sqsubset)$ の最小元を表す．

```
class PairLL L {
  (Bool, L) first; (Bool, L) second;

  (Null, L) setFirst((Bool, L) fst) L {
    this.first = fst; result = null;
  }
  (Null, L) setSecond((Bool, L) snd) L {
    this.second = snd; result = null;
  }
  (Bool, L) getFirst() H { result = this.first; }
  (Bool, L) getSecond() H { result = this.second; }
}
class PairLH L {
  (Bool, L) first; (Bool, H) second;

  (Null, L) setFirst((Bool, L) fst) L {
    this.first = fst; result = null;
  }
  (Null, L) setSecond((Bool, H) snd) H {
    this.second = snd; result = null;
  }
  (Bool, L) getFirst() H { result = this.first; }
  (Bool, H) getSecond() H { result = this.second; }
}
```

図 1　2 項組クラス PairLL と PairLH の実装例

3.1　機密度の表現と構文

プログラム中に型の一部として指定する機密度を機密度定数 η と機密度パラメータのリスト $\overline{X}$ の組で表す．直感的には機密度定数 η とすべての機密度パラメータ $\overline{X}$ の結びがそのデータの機密度であることを意味する．情報流解析では，あるデータの機密度はそのデータが依存するすべてのデータの機密度の上界のいずれかであるとみなす．機密度が機密度定数 η_1，η_2 のデータに依存するデータの機密度には，η_1 と η_2 の結びが求められるため，結びである機密度定数を指定すれば良い．しかし，機密度が機密度パラメータ X_1，X_2 のデータに依存するデータの機密度は X_1 以上かつ X_2 以上でなければならないが，それぞれが具体的な機密度定数でないため結びが求められない．そこで，プログラム中のデータの機密度を，依存するデータの機密度が機密度定数の場合はそれらの結びである機密度定数で，機密度パラメータの場合はそれらのリストで表現する．

図 2 に機密度がパラメータ化されたオブジェクト指向言語の構文を示す．図中の $\overline{A}$ は長さが 0 以上の有限リストを表す．機密度 ρ は機密度定数 η と機密度パラメータのリスト $\overline{X}$ の組である．簡単のために，機密度パラメータのリスト $\overline{X}$ の長さが 0 の時は単に η と書く．機密度定数 η が最小元 $\bot_{\mathcal{H}}$ で機密度パラメータのリスト $\overline{X}$ の長さが 1 のときは単に X と書く．τ は型情報であり，データ型 $T{<}\overline{\rho}{>}$ と機密度 ρ の組である．$T{<}\overline{\rho}{>}$ は T の機密度パラメータ $\overline{X}$ に機密度 $\overline{\rho}$ が割り当てられた型である．データ型 $T{<}\overline{\rho}{>}$ の機密度のリスト $\overline{\rho}$ の長さが 0 のとき，単に T と書く．CL はクラス定義であり，0 個以上の機密度パラメータの宣言と 0 個以上のフィールド宣言，0 個以上のメソッド宣言からなる．CL の ρ はクラスの機密度を表す．M はメソッド宣言であり，戻り値の型とメソッド名 m，仮引数に加えて，メソッドのヒープエフェクトのリスト $\overline{\rho}$ が指定される．ヒープエフェクトはインスタンス生成や

$$
\begin{array}{rcl}
T & ::= & \mathrm{Bool} \mid \mathrm{Null} \mid C \\
\tau & ::= & (T\!<\!\overline{\rho}\!>,\ \rho) \\
\rho & ::= & \eta(\overline{X}) \\
CL & ::= & \mathrm{class}\ C\!<\!\overline{X}\!>\ \rho\ \mathrm{extends}\ C\!<\!\overline{\rho}\!>\ \{\overline{\tau\ f};\ \overline{M}\} \\
M & ::= & \tau\ m(\overline{\tau\ x})\ \overline{\rho}\ B \\
B & ::= & \{\overline{\tau\ x};\ \overline{S};\} \\
S & ::= & x\ =\ e \mid e.f = e \mid \mathrm{if}\ (e)\ B\ \mathrm{else}\ B \mid x\ =\ e.m(\overline{e}) \mid x\ =\ \mathrm{new}\ C\!<\!\overline{\rho}\!> \\
e & ::= & x \mid e.f \mid \mathrm{true} \mid \mathrm{false} \mid \mathrm{null} \mid e\ ==\ e
\end{array}
$$

図 2　提案する言語の構文

$$
\frac{\eta_1 \sqsubseteq \eta_2 \quad \overline{X_1} \subseteq \overline{X_2}}{\eta_1(\overline{X_1}) \preceq \eta_2(\overline{X_2})}\ [\text{REL-RHO}]
$$

図 3　ρ の関係

フィールド代入によって変更されるヒープ上のデータの機密度である．対象とする言語ではインスタンスはヒープ上に生成される．データの機密度がすべて機密度定数であればそれらの交わりをヒープエフェクトとすればよいが，機密度パラメータが指定される場合もあるため交わりを求められない．そこで，メソッドのヒープエフェクトには変更されるヒープ上の各データの機密度を列挙する．機密度パラメータに機密度定数が割り当てられた場合，それらの交わりが具体的なヒープエフェクトである．B はブロックを表し，0 個以上のローカル変数宣言と 0 個以上の文からなる．S と e はそれぞれ文と式である．

3.2　機密度の大小関係

機密度 ρ の大小関係 $\preceq$ の定義を図 3 に示す．ここで，$\overline{X_1} \subseteq \overline{X_2}$ は $\overline{X_1}$ と $\overline{X_2}$ それぞれについて重複する要素を除いて集合とみなした時の包含関係である．機密度 ρ_1 と ρ_2 の機密度パラメータに何らかの機密度束から機密度定数を割り当てて求められるそれぞれの結びである機密度定数を η_1' および η_2' とする．この時，機密度パラメータにいかなる機密度定数が割り当てられたとしても，$\rho_1 \preceq \rho_2$ ならば $\eta_1' \sqsubseteq \eta_2'$ となるように $\preceq$ は定義される．$\overline{X_1} \subseteq \overline{X_2}$ の時，$\overline{X_1}$ と $\overline{X_2}$ それぞれに対し任意の機密度定数のリスト $\overline{\eta_\alpha}$ と $\overline{\eta_\beta}$ が割り当てられると，$\overline{\eta_\alpha} \subseteq \overline{\eta_\beta}$ より $\bigsqcup \overline{\eta_\alpha} \sqsubseteq \bigsqcup \overline{\eta_\beta}$ であるため機密度パラメータのリストはリストを集合とみなしたときの包含関係 $\subseteq$ で比較する．ここで，$\eta \sqcup \eta'$ は η と η' の結びである機密度定数を表し，$\bigsqcup \overline{\eta}$ はすべての η の結びである機密度定数を表す．

3.3　型付け規則

図 4 に提案する言語の型付け規則を示す．CDEC 規則がクラス宣言，MDEC 規則がメソッド宣言の型付け規則で，E-で始まる規則は式の型付け規則である．それ以外の規則は文の型付け規則である．型付け可能なプログラムは機密度パラメータにいかなる機密度定数が割り当てられて具体化されたとしても非干渉性を満たすように型付け規則を定める．$Set(\overline{x})$ は $\overline{x}$ から重複する要素を除外し，集合を生成する関数である．

データ型 $T\!<\!\overline{\rho}\!>$ のサブタイプ関係 $\trianglelefteq$ は以下を満たすように定める．

- $\forall T\!<\!\overline{\rho}\!>.T\!<\!\overline{\rho}\!> \trianglelefteq T\!<\!\overline{\rho}\!>$
- $\forall T\!<\!\overline{\rho}\!>.Null \trianglelefteq T\!<\!\overline{\rho}\!>$
- $C\!<\!\overline{\rho_C}\!> \trianglelefteq D\!<\!\overline{\rho_D}\!>$ iff $C = D \wedge \overline{\rho_C = \rho_D}$ または C の宣言が $\mathrm{class}\ C\!<\ \overline{X}\ >$ $\mathrm{extends}\ F\ <\ \overline{\rho_F}\ >\ \{...\}$ の時 $F\!<\![\overline{\rho_C}/\overline{X}]\rho_F\!> \trianglelefteq D\!<\!\overline{\rho_D}\!>$

$$\frac{\begin{array}{l} glevel(D{<}\overline{\rho_{gD}}{>}) \preceq \rho \quad C{<}\overline{X}{>}\ \ \rho \text{ extends } D{<}\overline{\rho_{gD}}{>} \vdash M \text{ for each } M \in \overline{M} \\ glevel(D{<}\overline{\rho_{gD}}{>}) \neq \rho \Rightarrow \\ \quad \text{every } m \text{ with } gsmtype(m, D{<}\overline{\rho_{gD}}{>}) \text{ defined is overridden in } C \end{array}}{\vdash C{<}\overline{X}{>}\ \ \rho \text{ extends } D{<}\overline{\rho_{gD}}{>}\ \ \{\overline{\tau\ f};\ \overline{M}\}} \text{ [CDEC]}$$

$$\frac{\begin{array}{l} \overline{x : \tau_x},\ this : (C{<}\overline{X}{>}, \rho),\ result : \tau_r \vdash B : (\rho_s, P) \\ gsmtype(m, D{<}\overline{\rho_{gD}}{>}) \text{ is defined } \Rightarrow gsmtype(m, D{<}\overline{\rho_{gD}}{>}) = \overline{\tau_x} \xrightarrow{\overline{\rho_h}} \tau_r \\ \forall \rho_1 \in P. \exists \rho_2 \in \overline{\rho_h}. \rho_2 \preceq \rho_1 \end{array}}{C{<}\overline{X}{>}\ \ \rho \text{ extends } D{<}\overline{\rho_{gD}}{>} \vdash \tau_r\ m(\overline{\tau_x\ x})\ \overline{\rho_h}\ B} \text{ [MDEC]}$$

$$\frac{\begin{array}{l} \Delta,\ \overline{x : (T_x{<}\overline{\rho_{gx}}{>}, \rho_x)} \vdash S_i\ :\ (\rho_{s_i},\ P_i) \\ \rho_s \preceq \rho_{s_i} \quad P = \bigcup_i P_i \end{array}}{\Delta \vdash \{\overline{(T_x{<}\overline{\rho_{gx}}{>}, \rho_x)\ x};\ S_1; \ldots S_n;\}\ :\ (\rho_s,\ P)} \text{ [BLOCK]}$$

$$\frac{\begin{array}{l} \Delta \vdash x : (T_x{<}\overline{\rho_{gx}}{>}, \rho_x) \quad \Delta \vdash e : (T_e{<}\overline{\rho_{ge}}{>}, \rho_e) \\ T_e{<}\overline{\rho_{ge}}{>} \trianglelefteq T_x{<}\overline{\rho_{gx}}{>} \quad \rho_e \preceq \rho_x \quad \rho_s \preceq \rho_x \end{array}}{\Delta \vdash x = e\ :\ (\rho_s,\ \emptyset)} \text{ [ASSIGN1]}$$

$$\frac{\begin{array}{l} \Delta \vdash e_1 : (T_1{<}\overline{\rho_{g1}}{>}, \rho_1) \quad \Delta \vdash e_2 : (T_2{<}\overline{\rho_{g2}}{>}, \rho_2) \\ (T_f{<}\overline{\rho_{gf}}{>}, \rho_f)\ f \in gsfields(T_1{<}\overline{\rho_{g1}}{>}) \\ T_2{<}\overline{\rho_{g2}}{>} \trianglelefteq T_f{<}\overline{\rho_{gf}}{>} \quad \rho_1 \preceq \rho_f \quad \rho_2 \preceq \rho_f \end{array}}{\Delta \vdash e_1.f = e_2\ :\ (\rho_s,\ \{\rho_f\})} \text{ [ASSIGN2]}$$

$$\frac{\begin{array}{l} \Delta \vdash x : (T_x\ {<}\overline{\rho_{gx}}{>}, \rho_x) \quad \Delta \vdash e : (T_e{<}\overline{\rho_{ge}}{>}, \rho_e) \quad \Delta \vdash e_i : (T_{e_i}\ {<}\overline{\rho_{ge_i}}{>}, \rho_{e_i}) \\ (T_{y_1}{<}\overline{\rho_{gy_1}}{>}, \rho_{y_1}), \ldots, (T_{y_n}{<}\overline{\rho_{gy_n}}{>}, \rho_{y_n}) \xrightarrow{\overline{\rho'_h}} (T_r{<}\overline{\rho_{gr}}{>}, \rho_r) \\ \quad = gsmtype(m, T_e{<}\overline{\rho_{ge}}{>}) \\ \forall i \in \{1, \ldots, n\}.\ T_{e_i}{<}\overline{\rho_{e_i}}{>} \trianglelefteq T_{y_i}{<}\overline{\rho_{y_i}}{>},\ \rho_{e_i} \preceq \rho_{y_i} \\ T_r{<}\overline{\rho_{gr}}{>} \trianglelefteq T_x{<}\overline{\rho_{gx}}{>} \quad \rho_e \preceq \rho_x \quad \rho_r \preceq \rho_x \quad \rho_s \preceq \rho_x \\ P = Set(\overline{\rho'_h}) \quad \forall \rho_h \in P. \rho_e \preceq \rho_h \end{array}}{\Delta \vdash x = e.m(e_1, \ldots, e_n) : (\rho_s, P)} \text{ [CALL]}$$

$$\frac{\Delta \vdash x : (T_x{<}\overline{\rho_{gx}}{>}, \rho_x) \quad C{<}\overline{\rho_C}{>} \trianglelefteq T_x{<}\overline{\rho_{gx}}{>} \quad glevel(C{<}\overline{\rho_C}{>}) \preceq \rho_x \quad \rho_s \preceq \rho_x}{\Delta \vdash x = \text{new } C{<}\overline{\rho_C}{>} : (\rho_s, \{glevel(C{<}\overline{\rho_C}{>})\})} \text{ [NEW]}$$

$$\frac{\begin{array}{l} \Delta \vdash e : (Bool, \rho_e) \quad \Delta \vdash B : (\rho_1, P) \quad \Delta \vdash B' : (\rho'_1, P') \\ \rho_e \preceq \rho_s \quad \rho_s \preceq \rho_1 \quad \rho_s \preceq \rho'_1 \quad \forall \rho_h \in P \cup P'. \rho_e \preceq \rho_h \end{array}}{\Delta \vdash \text{if } (e)\ B \text{ else } B' : (\rho_s, P \cup P')} \text{ [IF]}$$

$$\frac{(T_x{<}\overline{\rho_{gx}}{>}, \rho_x) = \Delta(x)}{\Delta \vdash x : (T_x{<}\overline{\rho_{gx}}{>}, \rho_x)} \text{ [E-VAR]} \qquad \frac{\begin{array}{l} \Delta \vdash e\ :\ (T_1{<}\overline{\rho_{g1}}{>}, \rho_1) \\ (T_f{<}\overline{\rho_{gf}}{>}, \rho_f)\ f \in gsfields(T_1{<}\overline{\rho_{g1}}{>}) \\ \rho_1 \preceq \rho_e \quad \rho_f \preceq \rho_e \end{array}}{\Delta \vdash e.f : (T_f{<}\overline{\rho_{gf}}{>}, \rho_e)} \text{ [E-FIELD]}$$

$$\frac{}{\Delta \vdash \text{true} : (\text{Bool}, \bot_{\mathcal{H}})} \text{ [E-TRUE]} \qquad \frac{}{\Delta \vdash \text{false} : (\text{Bool}, \bot_{\mathcal{H}})} \text{ [E-FALSE]}$$

$$\frac{}{\Delta \vdash \text{null} : (\text{Null}, \bot_{\mathcal{H}})} \text{ [E-NULL]} \qquad \frac{\begin{array}{l} \Delta \vdash e_1\ :\ (T{<}\overline{\rho_g}{>}, \rho_1) \quad \Delta \vdash e_2\ :\ (T{<}\overline{\rho_g}{>}, \rho_2) \\ \rho_1 \preceq \rho_e \quad \rho_2 \preceq \rho_e \end{array}}{\Delta \vdash e_1\ ==\ e_2 : (\text{Bool}, \rho_e)} \text{ [E-EQ]}$$

図 4　型付け規則

$$\frac{\text{class } C{<}\overline{X}{>}\ \ \rho_C \text{ extends } D{<}\overline{\rho}{>}\ \{...\}}{glevel(C{<}\overline{\rho_g}{>}) = [\overline{\rho_g}/\overline{X}]\rho_C}\ \text{[GLEVEL]}$$

$$\frac{\begin{array}{l}\text{class } C{<}\overline{X}{>}\ \ \rho_C \text{ extends } D{<}\overline{\rho}{>}\ \{\overline{\tau\ x};\ \overline{M}\}\\ \tau\ m(\overline{\tau\ u})\ \overline{\rho_h}\ B \in \overline{M}\end{array}}{gsmtype(m, C{<}\overline{\rho_g}{>}) = [\overline{\rho_g}/\overline{X}](\overline{\tau} \xrightarrow{\overline{\rho_h}} \tau)}\ \text{[GSMTYPE-IN-C]}$$

$$\frac{\begin{array}{l}\text{class } C{<}\overline{X}{>}\ \ \rho_C \text{ extends } D{<}\overline{\rho}{>}\ \{\overline{\tau\ x};\ \overline{M}\}\\ \tau\ m(\overline{\tau\ u})\ \overline{\rho_h}\ B\ \not\in \overline{M}\\ \overline{\tau} \xrightarrow{\overline{\rho_h}} \tau = gsmtype(m, D{<}\overline{\rho}{>})\end{array}}{gsmtype(m, C{<}\overline{\rho_g}{>}) = [\overline{\rho_g}/\overline{X}](\overline{\tau} \xrightarrow{\overline{\rho_h}} \tau)}\ \text{[GSMTYPE-NOT-IN-C]}$$

$$\frac{\text{class } C{<}\overline{X}{>}\ \ \rho_C \text{ extends } D{<}\overline{\rho}{>}\ \{\overline{\tau\ f};\ \overline{M}\}}{gsfields(C{<}\overline{\rho_g}{>}) = (\overline{[\overline{\rho_g}/\overline{X}]\tau\ f}) \cup gsfields(D{<}\overline{[\overline{\rho_g}/\overline{X}]\rho}{>}))}\ \text{[GSFIELDS]}$$

図 5　glevel，gsmtype，gsfields の定義

文 S の型判定式 $\Delta \vdash S : (\rho_s, P)$ は型環境 Δ のもとで，以下を満たすことを表す．

- S で代入される変数や呼び出すメソッドの引数の機密度は ρ_s 以上
- すべての $\rho_h \in P$ について，S によるヒープエフェクトは ρ_h 以上
- S 内の情報流が安全である

(ρ_s, P) が文の型であり，P は文のヒープエフェクトの集合である．型環境 Δ は変数名から変数の型 τ を返す関数である．S 内の情報流が安全であるとは，x から y への情報流が存在するとき，x の機密度 ρ_x と y の機密度 ρ_y は $\rho_x \preceq \rho_y$ を満たすことを指す．

式 e の型判定式 $\Delta \vdash e : (T{<}\overline{\rho}{>}, \rho)$ は型環境 Δ のもとで，以下を満たすことを表す．

- e の値のデータ型は $T{<}\overline{\rho}{>}$ のサブタイプ
- e のすべての部分式の機密度は ρ 以下

図 4 内の関数 glevel，gsmtype，gsfields の定義を図 5 に示す．$glevel(T{<}\overline{\rho}{>})$ はクラス T の機密度を返す関数である．$gsmtype(m, T{<}\overline{\rho}{>})$ はクラス T もしくはその基底クラスで定義されるメソッド m の型を返す関数である．$gsfields(T{<}\overline{\rho}{>})$ はクラス T のフィールド宣言の集合を返す関数である．いずれの関数も返す値の機密度内の T の機密度パラメータ $\overline{X}$ を $\overline{\rho}$ で置換する．

図 5 内の機密度パラメータの置換規則 $[\rho_1, \ldots, \rho_n/Y_1, \ldots, Y_n]\rho$，$[\overline{\rho_g}/\overline{X}](\overline{\tau} \xrightarrow{\overline{\rho_h}} \tau)$，$[\overline{\rho_g}/\overline{X}]\tau$ の定義を図 6 に示す．$[\rho_1, \ldots, \rho_n/Y_1, \ldots, Y_n]\rho$ は $\rho = \eta(\overline{X})$ の $\overline{X}$ に含まれるすべての Y_i を ρ_i で置き換えた機密度である．置換後の機密度の機密度定数は η と置換された Y_i に対応する ρ_i の機密度定数の結びである機密度定数とする．$Y_1, \ldots, Y_n$ に含まれない $\overline{X}$ の要素のリストと置換された Y_i に対応する ρ_i の機密度パラメータのリストを連結したリストが置換後の機密度の機密度パラメータのリストである．$[\overline{\rho_g}/\overline{X}](\overline{\tau} \xrightarrow{\overline{\rho_h}} \tau)$ はメソッドの型に対する機密度パラメータの置換規則であり，引数のすべての型，戻り値の型，ヒープエフェクトに対して機密度パラメータの置換を行う．$[\overline{\rho_g}/\overline{X}]\tau$ はデータ型の機密度パラメータと機密度内の機密度パラメータの置換を行う．

MDEC 規則はメソッド宣言の規則である．メソッド呼び出しによってレシーバの機密度よりも低いヒープエフェクトが発生してはいけないため，メソッドのヒープエフェクトがメソッドの処理によって影響を及ぼされるヒープ上のデータの機密度の下界であることを検査する．そこで，引数 $\overline{x : \tau_x}$ や戻り値 $result : \tau_r$，自インスタ

$$\frac{I = \{i \mid Y_i \in \overline{X}\} \quad \eta' = \eta \sqcup (\bigsqcup_{i \in I} \eta_i) \quad \overline{X'} = \{X \in \overline{X} \mid X \notin \{Y_1, \ldots, Y_n\}\} \cup (\bigcup_{i \in I} \overline{Z_i})}{[\eta_1(\overline{Z_1}), \ldots, \eta_n(\overline{Z_n})/Y_1, \ldots, Y_n]\eta(\overline{X}) \;=\; \eta'(\overline{X'})} \text{ [REPLACE]}$$

$$\frac{}{[\overline{\rho_g}/\overline{X}](T\!<\!\overline{\rho}\!>,\, \rho) \;=\; (T\!<\!\overline{[\overline{\rho_g}/\overline{X}]\rho}\!>,\, [\overline{\rho_g}/\overline{X}]\rho)} \text{ [REPLACE-TAU]}$$

$$\frac{}{[\overline{\rho_g}/\overline{X}](\tau \xrightarrow{\overline{\rho_h}} \tau) \;=\; (\overline{[\overline{\rho_g}/\overline{X}]\tau} \xrightarrow{\overline{[\overline{\rho_g}/\overline{X}]\rho_h}} [\overline{\rho_g}/\overline{X}]\tau)} \text{ [REPLACE-GSMTYPE]}$$

図 6　機密度パラメータの置換規則の定義

```
class Pair<F, S> L {
  (Bool, F) first; (Bool, S) second; (Bool, L(F, S)) same;

  (Bool, F) getFirst() H { result = this.first; }
  (Bool, S) getSecond() H { result = this.second; }
  (Null, L) set((Bool, F) fst, (Bool, S) snd) F, S {
    (Null, L) x;
    this.first = fst; this.second = snd;
    x = this.updateSame(); result = null;
  }
  (Null, L) updateSame() L(F, S) {
    this.same = this.first == this.second; result = null;
  }
  (Bool, L(F, S)) isSame() H { result = same; }
}
```

図 7　機密度がパラメータ化された 2 項組クラスの実装例

ンスへの参照 $this : (C\!<\!\overline{X}\!>, \rho)$ を型環境とした下で，メソッド宣言のブロック B の型が (ρ_s, P) となる時，ヒープエフェクトの集合 P のそれぞれのヒープエフェクト ρ_1 について，ρ_1 以下となる機密度がメソッドのヒープエフェクトのリスト $\overline{\rho_h}$ に存在すれば型付け可能とする．メソッド呼び出し文の規則である CALL 規則では，呼び出すメソッド m のすべての実引数 e_i の機密度 ρ_{e_i} が対応する仮引数の機密度 ρ_{y_i} 以下であり，戻り値の機密度 ρ_r が代入先変数の機密度 ρ_x 以下であることを求める．レシーバの機密度よりも低いヒープエフェクトが生じないように，呼び出すメソッドのヒープエフェクトのリストそれぞれについてレシーバの機密度以上であれば型付け可能とする．メソッド呼び出し文の型は代入先変数の機密度以下の機密度と呼び出すメソッドのヒープエフェクトの集合の組である．

4　適用例

図 7 に Bool の 2 項組を表すクラス Pair の定義を示す．Pair には 2 項組の要素 first と second のアクセサメソッドに加えて，first と second が同値かを表すフィールド same とメソッド isSame がある．same の値はセッタメソッドで呼び出される updateSame メソッドで更新される．機密度パラメータ F は要素 first の機密度を表し，機密度パラメータ S は要素 second の機密度を表す．機密度束は $(\{L, H\}, \sqsubseteq)$ とし，$L \sqsubseteq H$ を満たすとする．

$$\dfrac{\dfrac{\Delta \vdash S_1 : (L, \{F\}) \quad \Delta \vdash S_2 : (L, \{S\}) \quad \Delta \vdash S_3 : (L, \{L(F,S)\}) \quad \Delta \vdash S_4 : (L, \emptyset)}{\Delta' \vdash \{ \; (Null, L)x; S_1; S_2; S_3; S_4; \; \} : (L, \{F, S, L(F,S)\})}\,[\text{BLOCK}] \quad \forall \rho \in \{F, S, L(F,S)\}.\exists \rho_h \in \{F, S\}.\rho_h \preceq \rho}{\begin{array}{l}\text{class } Pair\!<\!F, S\!> L \vdash \\ (Null, L)\ set((Bool, F)\ fst,\ (Bool, S)\ snd)\ F, S\ \{ \\ \quad (Null, L)x; S_1; S_2; S_3; S_4; \\ \}\end{array}}\,[\text{MDEC}]$$

図 8　set の導出木

$$\dfrac{\dfrac{(Pair\!<\!F, S\!>, L) = \Delta(this)}{\Delta \vdash this : (Pair\!<\!F, S\!>, L)}\,[\text{E-VAR}] \quad \dfrac{(\text{Bool}, F) = \Delta(fst)}{\Delta \vdash fst : (\text{Bool}, F)}\,[\text{E-VAR}] \quad (\text{Bool}, F) first \in gsfields(Pair\!<\!F, S\!>) \quad \text{Bool} \trianglelefteq \text{Bool} \quad L \preceq F \quad F \preceq F}{\Delta \vdash this.first = fst : (L, \{F\})}\,[\text{ASSIGN-2}]$$

$$\dfrac{\dfrac{(Pair\!<\!F, S\!>, L) = \Delta(this)}{\Delta \vdash this : (Pair\!<\!F, S\!>, L)}\,[\text{E-VAR}] \quad \dfrac{(\text{Bool}, S) = \Delta(snd)}{\Delta \vdash snd : (\text{Bool}, S)}\,[\text{E-VAR}] \quad (\text{Bool}, S) second \in gsfields(Pair\!<\!F, S\!>) \quad \text{Bool} \trianglelefteq \text{Bool} \quad L \preceq S \quad S \preceq S}{\Delta \vdash this.second = snd : (L, \{S\})}\,[\text{ASSIGN-2}]$$

$$\dfrac{\dfrac{(\text{Null}, L) = \Delta(x)}{\Delta \vdash x : (\text{Null}, L)}\,[\text{E-VAR}] \quad \dfrac{(Pair\!<\!F, S\!>, L) = \Delta(this)}{\Delta \vdash this : (Pair\!<\!F, S\!>, L)}\,[\text{E-VAR}] \quad () \xrightarrow{L(F,S)} (\text{Null}, L) = gsmtype(updateSame, Pair\!<\!F, S\!>) \quad \text{Null} \trianglelefteq \text{Null} \quad L \preceq L \quad L \preceq L \quad L \preceq L \quad \forall \rho_h \in \{L(F,S)\}.L \preceq \rho_h}{\Delta \vdash x = this.updateSame() : (L, \{L(F,S)\})}\,[\text{CALL}]$$

$$\dfrac{\dfrac{(\text{Null}, L) = \Delta(result)}{\Delta \vdash result : (\text{Null}, L)}\,[\text{E-VAR}] \quad \dfrac{}{\Delta \vdash \text{null} : (\text{Null}, L)}\,[\text{E-NULL}] \quad \text{Null} \trianglelefteq \text{Null} \quad L \preceq L \quad L \preceq L}{\Delta \vdash result = \text{null} : (L, \emptyset)}\,[\text{ASSIGN-1}]$$

図 9　set の文 S_1 から S_4 の導出木

`Pair` は提案する型付け規則で型付け可能である．例として `set` メソッドの導出木を図 8 に示す．図 8 内の Δ' は $fst : (\text{Bool}, F), snd : (\text{Bool}, S), this : (Pair < F, S > , L), result : (\text{Null}, L)$ で Δ は $\Delta', x : (\text{Null}, L)$ である．S_1 と S_2，S_3，S_4 はそれぞれ $this.first = fst$ と $this.second = snd$，$x = this.updateSame()$，$result = null$ であり，それぞれ $(L, \{F\})$，$(L, \{S\})$，$(L, \{L(F,S)\})$，$(L, \emptyset)$ に型付けできる．S_1 から S_4 の導出木を図 9 に示す．メソッド `set` の処理はフィールドへの代入やメソッド呼び出しによって機密度が F，S，$L(F,S)$ のヒープ上のデータに影響を与える．いずれの機密度もメソッド `set` のヒープエフェクトである F 以上か S 以上であるため，型付け可能である．

`Pair` を利用するクラス `C<F>`を図 10 に示す．メソッド `method` は機密度が機密度パラメータ `F` の値と L の値の 2 項組 `pair` を引数に取り，2 つの値が同じときはク

```
class C<F> L {
  (C<F>, L) method((Pair<F, L>, L) pair) H {
    (Bool, F) isSame; isSame = pair.isSame();
    if (isSame) { result = this; } else { result = null; }
  }
}
```

図 10　Pair を用いるクラスの例

$$\dfrac{\dfrac{(\mathrm{Bool}, F) = \Delta(isSame)}{\Delta \vdash isSame : (\mathrm{Bool}, F)}\,[\text{E-VAR}] \qquad \Delta \vdash B : (L, \emptyset)\Delta \vdash B' : (L, \emptyset)}{F \preceq \rho_s \quad \rho_s \preceq L \quad \rho_s \preceq L \quad \forall \rho_h \in \emptyset \cup \emptyset . F \preceq \rho_h}\,[\text{IF}]$$
$$\Delta \vdash \text{if } (isSame)\ B \text{ else } B' : (\rho_s, \emptyset)$$

図 11　method の導出木

ラス C のインスタンスを返し，そうでないときは null を返すメソッドである．メソッド method の戻り値の機密度は L として定義されており，機密なデータではない．機密度パラメータ F が機密度定数 H で具体化された場合，引数 pair の要素 first は機密なデータである．しかし，引数 pair の second の値を変えてメソッドを呼び出し，機密でない戻り値を観測することで機密なデータを推測できるため，不正な情報流が存在する．図 11 に method の if 文が提案する型付け規則で型付けできないことを示す導出木を示す．図 11 の Δ は $pair : (Pair< F, L >, L), isSame : (\mathrm{Bool}, F), this : (C< F >, L), result : (C< F >, L)$ であり，B と B' はそれぞれ $\{result = this;\}$ と $\{result = \mathrm{null};\}$ である．B と B' の型はそれぞれ $(L, \emptyset)$ であるが，導出木は省略する．if 文の型 $(\rho_s, \emptyset)$ について $F \preceq \rho_s$ と $\rho_s \preceq L$ を満たす ρ_s は存在しないため，型付けできない．

5　関連研究

Agesen らや Bracha らはオブジェクト指向言語である Java 言語に型を引数に取るクラスやメソッドを導入した [6] [7]．五十嵐らは，Bracha らの言語と型付け規則を形式的に定義し，型付け規則が健全であることを証明した [8]．

情報流解析におけるプログラム中に指定する機密度をパラメータ化している言語には FlowCaml や JFlow，Jif がある [9] [10] [11]．FlowCaml は関数型言語である OCaml に対して情報流解析を追加した言語であり，JFlow，Jif はオブジェクト指向言語である Java 言語のサブセットに対して情報流解析を追加した言語である．Sun らは Banerjee らのオブジェクト指向プログラム向け情報流解析に対して機密度をパラメータ化したクラス定義を追加した [12]．FlowCalm や Sun らの言語ではプログラム中に指定する機密度は機密度定数か機密度パラメータのいずれか一つであり，機密度パラメータが満たすべき制約をメソッド宣言などに集合として列挙する．Sun らは機密度が付与されていないプログラムに対して自動的に機密度パラメータを追加し，機密度パラメータが満たすべき制約を推論し，プログラムに制約集合を付与するツールを提案している．機密度が機密度パラメータのデータに依存する変数ごとに機密度パラメータが必要になるため，プログラム中の機密度パラメータの数が多くなり，プログラム中に記述された制約集合から機密度パラメータ間の関係を開発者が把握することが難しい．本研究ではプログラム中の機密度に複数の機密度パラメータを指定でき，プログラム中の機密度パラメータの数が少なくなる．プログラム中に機密度パラメータが満たすべき制約の集合を記述する必要もない．

6 おわりに

本稿では，オブジェクト指向言語を対象とする情報流解析において，機密度のパラメータ化を提案した．クラス定義において機密度パラメータを宣言できる．データの機密度を機密度定数と機密度パラメータのリストの組として表現し，機密度間の大小関係を機密度定数間の関係と機密度パラメータのリストの包含関係に基づいて定義する．機密度定数と機密度パラメータのリストの結びがそのデータの機密度であることを意味する．機密度パラメータがいかなる機密度定数で具体化されても，型付け可能なプログラムは不正な情報流を含まないように型付け規則を定めた．例として，任意の機密度の 2 項組を表すクラスの定義とそのクラスを利用するプログラムを示し，機密度パラメータに割り当てられる機密度定数によっては不正な情報流が発生するプログラムは型付けできないことを示した．

本稿で提案する型付け規則が非干渉性に対して健全であることの証明は今後の課題である．本稿で提案する言語の表現力を確認し Sun らの言語の表現力との比較することも今後の課題である．本稿の型付け規則は，機密度パラメータにどのような機密度定数を割り当てても不正な情報流が存在しないことを保証しようとしている．そのために制約が厳しく，実用的なプログラムの実現を困難にしている可能性がある．

謝辞 本研究の一部は JSPS 科研費 15K00112, 17K12666 および 2017 年度南山大学パッヘ研究奨励金 I-A-2 の助成による．

参考文献

[1] Dennis Volpano, Cynthia Irvine, and Geoffrey Smith. A Sound Type System for Secure Flow Analysis. *J. Comput. Secur.*, Vol. 4, No. 2-3, pp. 167–187, 1996.

[2] Anindya Banerjee and David A. Naumann. Secure Information Flow and Pointer Confinement in a Java-like Language. In *Proceedings of the 15th IEEE Computer Security Foundations Workshop*, pp. 253–267. IEEE Computer Society Press, 2002.

[3] Andrei Sabelfeld and Andrew C. Myers. Language-Based Information-Flow Security. *IEEE Journal on Selected Areas in Communications*, Vol. 21, No. 1, pp. 5–19, 2003.

[4] 黒川翔, 桑原寛明, 山本晋一郎, 坂部俊樹, 酒井正彦, 草刈圭一朗, 西田直樹. 例外処理付きオブジェクト指向プログラムにおける情報流の安全性解析のための型システム. 電子情報通信学会論文誌, Vol. 91, No. 3, pp. 757–770, mar 2008.

[5] Dorothy E. Denning. A Lattice Model of Secure Information Flow. *Commun. ACM*, Vol. 19, No. 5, pp. 236–243, 1976.

[6] Ole Agesen, Stephen N. Freund, and John C. Mitchell. Adding type parameterization to the java language. In *Proceedings of the 12th ACM SIGPLAN Conference on Object-oriented Programming, Systems, Languages, and Applications*, OOPSLA '97, pp. 49–65, New York, NY, USA, 1997. ACM.

[7] Gilad Bracha, Martin Odersky, David Stoutamire, and Philip Wadler. Making the future safe for the past: Adding genericity to the java programming language. In *Proceedings of the 13th ACM SIGPLAN Conference on Object-oriented Programming, Systems, Languages, and Applications*, OOPSLA '98, pp. 183–200, New York, NY, USA, 1998. ACM.

[8] Atsushi Igarashi, Benjamin C. Pierce, and Philip Wadler. Featherweight java: A minimal core calculus for java and gj. *ACM Trans. Program. Lang. Syst.*, Vol. 23, No. 3, pp. 396–450, May 2001.

[9] Vincent Simonet. The Flow Caml System: documentation and user's manual. Technical Report 0282, Institut National de Recherche en Informatique et en Automatique (INRIA), July 2003. ©INRIA.

[10] Andrew C. Myers. JFlow: Practical Mostly-static Information Flow Control. In *Proceedings of the 26th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL '99, pp. 228–241, 1999.

[11] Jif: Java + information flow. `http://www.cs.cornell.edu/jif/`.

[12] Qi Sun, Anindya Banerjee, and David A Naumann. Modular and constraint-based information flow inference for an object-oriented language. In *SAS*, Vol. 3148, pp. 84–99. Springer, 2004.