3T-2

議論フレームワークにおけるideal意味論の解集合プログラミングによる計算

坂倉 広幸[*1]　　若木 利子[*2]
芝浦工業大学 システム工学部 電子情報システム学科[*3]

1 はじめに

議論において論証が正当化されるか否かを判定する方法として, 1995 年 Dung により議論フレームワーク (AF) と 4 種類の意味論が提案された [3]. 最近, grounded 意味論では余りにも慎重すぎて, 一意的に正当化される論証を見落としている問題点が指摘され, その問題を解決するために, 2007 年 Dung らにより ideal 意味論 [4] が提案された. ideal 意味論の extension は全ての preferred extension の共通部分の部分集合の中で極大な admissible set として定義される. ideal extension は grounded extension を包含し [4], grounded 意味論と同様 extension がユニークでありながら人間の sceptical な議論判定の考え方を grounded より適切に説明する. 他方, 2008 年, Dung の議論意味論を labelling で表す Caminada の方法 [2] と解集合プログラミングに基づいて議論意味論を計算する若木らの方法 [6] が提案され, その議論計算エンジンが開発された [7]. 本研究では, マルチエージェントシステムで ideal 意味論に基づく議論や交渉を可能とする為に, [6] と類似のアプローチで admissible labelling [1] と ideal set の性質に関する定理 [4] に基づき, ideal extension を解集合プログラミング (ASP) により計算する方法を提案し, 提案手法に基づく議論計算エンジンを開発した.

2 準備

2.1 議論フレームワークと ideal 意味論

議論フレームワークと Dung の議論意味論, 及び, 近年提案された ideal 意味論の定義を以下に示す.

定義 1 議論フレームワーク (以後, AF)[3] は (Ar, def) で定義される. 但し Ar は論証集合, def は Ar 上の関係 ($def \subseteq Ar^2$) である. $(a,b) \in def$ は,「a が b を攻撃する」ことを意味する. また, 論証集合 S, T について, $(a,b) \in def$ なる論証 $a \in S$, $b \in T$ が存在する時, 「S が T を攻撃する」という.

定義 2 (conflict-free/acceptable /admissible [3])
$AF = (Ar, def)$ と論証集合 $S \subseteq Ar$ について,

- S が conflict-free iff $\nexists\, a,\ b \in Ar$ s.t. $a\ def\ b$.
- 論証 a が S に関して acceptable iff $\exists c \in S$ s.t. $c\ def\ b$ for $\forall b \in Ar$ s.t. $b\ def\ a$.
- S が admissible iff S は conflict-free でかつ任意の論証 $a \in S$ が S に関して acceptable.

定義 3 (Dung の議論意味論 [3]) $AF = (Ar,\ def)$ が与えられている. $S \subseteq Ar$ を conflict-free な論証集合, 単調関数 $F : 2^{Ar} \to 2^{Ar}$を $F(S) = \{A | A$ は S に関して acceptable$\}$ と定義する. このとき complete/ preferred/ grounded/ stable の議論意味論は以下で定義する各 extension で与えられる.

- S が complete extension iff $S = F(S)$.
- S が preferred (resp. grounded) extension iff S は $\subseteq$ に関して極大 (resp. 極小) の complete extension.
- S が stable extension iff S は $Ar \backslash S$ の任意の論証を攻撃する preferred extension.

定義 4 (ideal 意味論 [4]) $AF = (Ar,\ def)$ において, ideal 意味論は以下の ideal extension で与えられる.

- S は ideal set iff S は admissible でかつ S は任意の preferred extension の部分集合.
- S が ideal extension iff S は ideal set の中で $\subseteq$ に関して極大.

例題 1 (ideal と grounded の違い) ideal 意味論と grounded 意味論とで extension が異なる例題を示す.

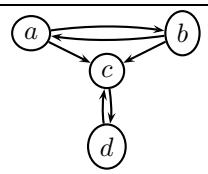

図 1 ideal と grounded で extension の異なる AF

議論内容:
c さん:行きたい国がないなら国内旅行にしよう.
a さん:僕はイギリスに行きたい!
b さん:俺はフランスがいい!
d さん:海外に行くなら, パスポートが必要だよ.

complete extension : $\emptyset,\ \{d\},\ \{a,\ d\},\ \{b,\ d\}$
preferred extension : $\{a,\ d\},\ \{b,\ d\}$
grounded extension : $\emptyset$
admissible : $\emptyset,\ \{a\},\ \{b\},\ \{d\},\ \{a,\ d\},\ \{b,\ d\}$
ideal :$\emptyset,\ \{d\}$　　ideal extension : $\{d\}$

この議題では人間の行う議論や思考では明らかにパスポートが必要という結論になるが, grounded extension は $\emptyset$ となり, 正当化される論証が存在しない. しかし, ideal extension では正当化論証は d となり, 人間の考え方と一致するので, ideal extension に優位性がある.

次の定理は文献 [4] の Theorem 3.3 で証明されている.

定理 1 どの admissible set からも攻撃されていない admissible set は, ideal であり, その逆も成り立つ.

2.2 Labelling

admissible set や各議論意味論の extension は Caminada の提案した labelling [1, 2] により表現される.

定義 5 (AF-Labelling [2]) $AF = (Ar,\ def)$ について AF-labelling $\mathcal{L}$ は全関数 $\mathcal{L} : Ar \to \{in,\ out,\ undec\}$ である. $in(\mathcal{L})$, $out(\mathcal{L})$, $undec(\mathcal{L})$ を以下で定義する.

- $in(\mathcal{L}) = \{a \in Ar | \mathcal{L}(a) = in\}$
- $out(\mathcal{L}) = \{a \in Ar | \mathcal{L}(a) = out\}$
- $undec(\mathcal{L}) = \{a \in Ar | \mathcal{L}(a) = undec\}$

定義 6 (admissible labelling[1]) 以下の条件を満たす AF-labelling を admissible labelling という.

1. if $\mathcal{L}(a) = in$ then $\forall b \in Ar$:$(b\ def\ a \supset \mathcal{L}(b) = out)$
2. if $\mathcal{L}(a) = out$ then $\exists b \in Ar$:$(b\ def\ a \wedge \mathcal{L}(b) = in)$

定理 2 [1] (Ar, def) の admissible labelling $\mathcal{L}$ と admissible set E につい以下の関係が成り立つ.

- admissible set $E = \{a \in Ar | \mathcal{L}(a) = in\}$

2.3 一般論理プログラムと解集合プログラミング

本研究では一般論理プログラム (以後, NLP) を対象とする. 解集合プログラミング (以後 ASP) とは論理プログラムを用いて問題の知識を表現し, その問題の解を解集合 [5] から得るプログラミング手法である.

定義 7 NLP は以下の形のルールの集合であり, 解集合 (安定モデル)[5] で意味論が与えられる.

$$A \leftarrow B_1,\ \cdots,\ B_m,\ not\ B_{m+1},\ \cdots,\ not\ B_n$$
$$\leftarrow B_1,\ \cdots,\ B_m,\ not\ B_{m+1},\ \cdots,\ not\ B_n$$

ここで, $n \geq m \geq 0$, A, B_i は原子式, not は NAF (Negation as Failure) 否定である.

Computing Ideal Semantics in Answer Set Programming
[*1] Hiroyuki Sakakura
[*2] Toshiko Wakaki
[*3] Shibaura Institute of Technology

3 ASP による ideal 意味論の計算

本研究の提案手法として, ideal 意味論の extension と 1 対 1 に解集合が対応する NLP の変換論理プログラムを提案, 解集合プログラミングにより, その NLP から ideal 意味論の extension を求める.

3.1 変換論理プログラムの構成

admissible labeling を以下の定義 8 の NLP Π_{Lab} で記述する。しかし定義 6 の admissible labeling は admissible set と多対 1 である. 本研究では 1 対 1 となるように, 定義 6 の条件 2 の if を if and only if (i.e. iff) に修正した admissible labeling を提案する. これは Π_{Lab} の *out* の原子式を頭部に持つルールで表現される.

定義 8 所与の $AF = (Ar,\ def)$ より, 以下の NLP $\Pi \stackrel{\text{def}}{=} \Pi_{AF} \cup \Pi_{Lab}$ を構成する.

- $\Pi_{AF} = \{ag(a) \leftarrow |a \in Ar\} \cup \{def(a,\ b) \leftarrow |(a,\ b) \in def\}$
- $\Pi_{Lab} = \{in(X) \leftarrow ag(X),\ not\ ng(X), not\ undec(X),$
  $ng(X) \leftarrow in(Y),\ def(Y,\ X),$
  $ng(X) \leftarrow undec(Y),\ def(Y,\ X),$
  $out(X) \leftarrow in(Y),\ def(Y,\ X),$
  $undec(X) \leftarrow ag(X), not\ in(X), not\ out(X)\}$

NLP Π について次の定理が成り立つ。以下で, $\mathcal{I} \stackrel{\text{def}}{=} \{in(a) \mid a \in Ar\}$ を用いる.

定理 3 Π の解集合 S が存在すれば, $E = \{a|in(a) \in S\}$ なる AF の admissible set E が存在. 逆に admissible set E が存在すれば, $S|_{\mathcal{I}} = \{in(a) \mid a \in E\}$ なる Π の解集合 S が存在する.

次に ideal extension を計算するための変換論理プログラム: $tr[Ar,\ def;\ ideal]$ を示す. 以下で, S を解集合, X を集合とした時, $S|_X \stackrel{\text{def}}{=} S \cap X$ で定義する. [6] の方法と同様, 関数 ψ と 2 引数述語記号 m_2 を用いる.

定義 9 $AS = \{S|S$ は Π の解集合 $\}$, $\xi = |\text{AS}|$, ψ は $\psi = AS \rightarrow \{1,\cdots,\xi\}$ なる全単射関数とする. $\psi(S) = j$ なる解集合 $S \in$ に対して j を S の sid number と呼ぶ.

本研究では文献 [6] と同様, 述語記号 m_2 を用いて,

$$in(a) \in S_j \text{ s.t. } \psi(S_j) = j \ \text{ iff } \ m_2(a, j) \in M \quad (1)$$

が成り立つように Π の任意の解集合 S_j $(1 \leq j \leq \xi)$ の情報を変換論理プログラムの解集合 M に埋め込む。これは変換論理プログラムの Γ のルールで表現される.

定義 10 所与の $AF = (Ar,\ def)$ から構成される ideal 意味論の変換論理プログラムを以下で定義する.

$$tr[Ar,\ def;\ ideal] \stackrel{\text{def}}{=} \Pi \cup \Gamma \cup \Xi$$

- 但し, Γ は領域依存の以下のルールの集合である。
  $\psi(S) = j$ $(1 \leq j \leq \xi)$ なる $\forall S \in AS$ について
  $m_2(a,\ j) \leftarrow . \qquad sid(j) \leftarrow .$
- Ξ は領域独立の以下のルールの集合である。
  1. $\leftarrow m_2(Y,\ N),\ def(Y,\ X),\ in(X).$
  2. $attacked(N) \leftarrow m_2(Y,\ M),\ def(Y,\ X),\ m_2(X,\ N).$
  3. $ideal(N) \leftarrow sid(N),\ not\ attacked(N).$
  4. $c(N) \leftarrow in(X),\ ideal(N),\ not\ m_2(X, N).$
  5. $d(N) \leftarrow ideal(N),\ m_2(X,\ N),\ not\ in(X).$
  6. $\leftarrow d(N),\ not\ c(N).$

以下で $tr[Ar, def; ideal] = \Pi \cup \Gamma \cup \Xi$ について説明する. M を $\Pi \cup \Gamma \cup \Xi$ の解集合とすると, $S = M \cap \mathcal{B}_\Pi$ (但し, $\mathcal{B}_\Pi$ は Π のエルブラン基底) は Π の解集合 (安定モデル) であり, $E_M = in(\mathcal{L}_M) = \{a|in(a) \in S|_{\mathcal{I}} = M|_{\mathcal{I}}\}$ なる admissible set E_M, admissible labelling $\mathcal{L}_M$ を表す. M はルール *1* の一貫性制約を充足しているので, $in(a) \in M$ について, $(b, a) \in def$, かつ $m_2(b, N) \in M$ (i.e. $in(b) \in S_N$) なる論証 $b \in Ar$ が存在しないことを意味する. 即ち admissible set E_M はいかなる admissible set $E_N = \{b \mid in(b) \in S_N\}(1 \leq N \leq \xi)$ からも攻撃されていないので, 定理 1 より ideal set である。ルール *2, 3* より, $ideal(N) \in M$ であるならば, (1) により Γ に m_2 のアトムとして埋め込まれた Π の解集合 S_N s.t. $\psi(S_N) = N$ に対する admissible set $E_N = \{b \mid in(b) \in S_N\}$ が定理 1 より ideal set であることを意味する. ルール *4, 5, 6* は [6] の手法を用いて, $\Pi \cup \Gamma \cup \Xi$ の解集合 M で表現された ideal set E_M が, $ideal(N) \in M$ なる全ての ideal set E_N に関して極大, 即ち, ideal extension か否かを判定する.

3.2 健全性/完全性定理と質問応答

変換論理プログラムについて以下の定理が成り立つ.

定理 4（健全性/完全性定理） E が $AF = (Ar, def)$ の ideal extension ならば, $M|_{\mathcal{I}} = \{in(a)|a \in E\}$ なる $tr[Ar, def; ideal]$ の解集合 M が存在する. 逆に, M が $tr[Ar, def; ideal]$ の解集合であるならば, $E = \{a|in(a) \in M\}$ なる AF の ideal extension E が存在する.

定理 5（質問応答） ideal 意味論の下で論証 $a \in Ar$ は,

- 正当化 (或いは, 却下) される. iff $tr[Ar,\ def;\ ideal] \cup \{\leftarrow not\ in(a)\}$(或いは, $\cup\{\leftarrow not\ out(a)\}$) が無矛盾.
- それ以外の場合, 引き分け (i.e. defensible).

4 実装・検証・評価

本研究の提案手法を ASP ソルバの DLV と C 言語を用いて実装した. さらに Java を用いて GUI インターフェースを持つツールとして実現し, 伊藤 ・ 龍沢らの議論計算エンジン [7, 8] も起動可能とした.

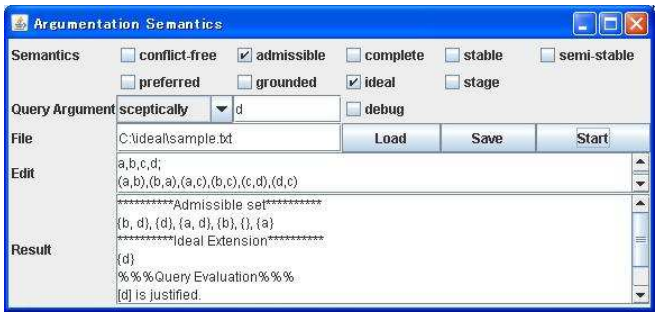

図 2 議論計算エンジンの GUI と例題 1 の実行例

5 おわりに

この議論計算エンジン及び GUI は現在 Windows 及び Linux 上にて稼働し, 本学の HP 上に近日公開予定である.

参考文献

[1] M. Caminada: A Labelling Approach for Ideal and Stage Semantics, Argument & Computation 2(1):1-21, 2011.

[2] M. Caminada: On the issue of reinstatement in argumentation, Proc. of JELIA-2006, pp.111-123, LNAI, Vol. 4160, Springer, 2006.

[3] P.M.Dung: On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-persons games, Artificial Intelligence **77**, pp.321-357,1995

[4] P.M. Dung , P. Mancarella, F. Toni : Computing ideal sceptical argumentation, Artificial Intelligence **171**, pp.642-674, 2007.

[5] Gelfond, M. and Lifschitz, V. Classical Negation in Logic Programs and Disjunctive Databases. New Generation Computing 9, pp.365-385, 1991.

[6] T.Wakaki and K.Nitta: Computing Argumentation Semantics in Answer Set Programming, New Frontiers in Artificial Intelligence, LNAI, Vol.5447, pp.254-269, Springer, 2008.

[7] 伊藤 浩太, 若木 利子: 解集合プログラミングによる議論の意味論の計算, 情報処理学会第 70 回全国大会講演論文集, 3V-9, pp.2-253-2-254, 2008.

[8] 龍沢昌宏, 若木利子: 議論フレームワークにおける stage 意味論の解集合プログラミングによる計算, 情報処理学会第 73 回全国大会, 1R-8, pp. 2-241 - 2-242, 2011.