日本ソフトウェア科学会第 27 回大会 (2010 年度) 講演論文集

脅威をもつ議論木に対する動的勝敗判定アルゴリズム

小野 司郎　高橋 和子

本研究は，議論を交わすことによって参加者の知識ベースが変更し新たな進行を見せるような現象を扱うような議論システムにおける勝敗の判定方法について述べたものである．著者らはこれまでにこのような議論システムを構築した．このシステムにおいては，議論を動的な観点からとらえ，ある時点における自分の発話が相手に有利な情報を与えるような関係を脅威関係として導入し，これが議論の勝敗を判定するときの鍵になることを示した．本発表では，議論の勝敗の判定アルゴリズムを提案し，判定結果とこのシステムにおける議論の実行結果の関係について論じる．

1　はじめに

議論は対話の一種であり，2 人が交互に言い合うことで進行する．一般の対話とちがい，両者の発話には主張とその根拠が含まれ，相手の発話内容を否定することでそれらを攻撃していくことによって進行することから高い論理性をもつ．そのため議論を論理的枠組みで扱うという考え方は以前からなされてきたが，議論を論理プログラミングと明確な対応付けを行い，議論の枠組みを定式化したのが Dung である [6]．一方，対話はエージェント間の情報あるいは知識の交換とみなせることから，議論は矛盾する知識を新たに受け取った場合の信念変更や信念改訂など人工知能の研究ともかかわりが深く，多くの人工知能研究者の興味もひいている [11][13][17]．議論システムについては，defeasible logic programming を使った研究 [7][8][9][12] やマルチエージェント間の交渉や説得など様々な分野への応用 [10][1][14][2]，さらに，Dung の示した accepted set の概念を拡張した研究 [5] など多くの研究が行われている．

このような議論システムでは，エージェントの知識ベースが与えられたときにどのような議論が可能か，どのような戦略をたてられるか，といった議論を静的にとらえたものが多く，実行によって環境や知識ベースが変化するような場合を考慮したものはほとんどない．しかし，現実の議論においては自分の発話が相手に新しい情報を与えることで相手の知識ベースの内容が変化していくことによってさまざまな現象が起こる．著者らは議論を交わすことによってエージェントの知識ベースが変更されるような知識ベースの変更を伴う議論システムを構築した [15]．このシステムの目的は実際の議論に近いモデルを提供し議論を動的に捉えることであった．そのためにキーとなった概念が「脅威」である．脅威は直観的には，自分の発話が相手に有利になる情報を与えることで自分に対する新たな攻撃をうむような発話間の関係であり，脅威による副作用まで考慮しないと議論の勝敗はわからない．これらを踏まえて，著者らは，議論を動的にとらえてシミュレートするアルゴリズムを提案し，動的勝敗の概念を与えた．さらに，動的勝敗の判定のみを行う簡単なアルゴリズムも提案した [16]．しかし，そこで扱ったモデルでは脅威関係に制約条件があり，脅威が一般に議論にどのような影響を与えているのかを考慮するには不十分なものだった．そこで本研究

An algorithm for judging a dynamic winner on an argumentation tree with a threat
Shiro ONO, 関西学院大学理工学部, School of Science and Technology, Kwansei Gakuin University.
Kazuko TAKAHASHI, 関西学院大学理工学部, School of Science and Technology, Kwansei Gakuin University.

では，脅威関係をさらに詳細に考慮し，勝敗を判定するアルゴリズムの有効範囲について述べる．

本論文は以下のように構成される．第 2 節では，議論や議論フレームワークなどの基本概念を示す．第 3 節では，知識ベースの変更を伴う議論について説明した後，議論をシミュレートするアルゴリズムを示し，動的勝敗について述べる．第 4 節では，動的勝敗判定アルゴリズについて論じる．最後に第 5 節では結論と今後の課題を述べる．

2 議論

2.1 議論フレームワーク

Dung [6] に基づき議論フレームワークを定義する．

Definition 1 (無矛盾) Ψ を命題論理の論理式の集合とする．$\psi \in \Psi$ および $\neg\psi \in \Psi$ を満たす原子論理式 ψ が存在しなければ，Ψ を 無矛盾 という．

各エージェント a の知識ベース $\mathbf{K}_\mathrm{a}$ は命題論理の有限集合である．$\mathbf{K}_\mathrm{a}$ は無矛盾とは限らず，また演繹に関して閉じている必要もない

Definition 2 (支持) Ψ を空でない論理式の集合，ψ を論理式とすると，$\phi, \phi \rightarrow \psi \in \Psi$ であれば，Ψ を ψ の支持 という．

Definition 3 (論証) $\mathbf{K}_\mathrm{a}$ をエージェント a の知識ベースとする．(Ψ, ψ) を a の論証という．ただし，Ψ は $\mathbf{K}_\mathrm{a}$ の空でない部分集合かまたは無矛盾な ψ の支持であり，$\psi \in \mathbf{K}_\mathrm{a}$ である．

論証 $A = (\Psi, \psi)$ に対して，Ψ を**根拠**，ψ を**主張**と呼び，それぞれ $Grounds(A)$，$Sentence(A)$ と表す．また， $Grounds(A) \cup \{Sentence(A)\}$ を $S(A)$ と表記する．$\mathbf{K}_\mathrm{a}$ から生成される可能性のあるすべての論証の集合を $AR_{\mathbf{K}_\mathrm{a}}$ と表す．

議論がループに陥ることを防ぐために**優先度** [3] [11] の概念を導入する．優先度は強さ，確からしさ，安定さなどを考慮して各論理式に対してあらかじめ付加されている評価値であり，これをもとに各論証の優先度を定める．

Definition 4 (攻撃) A_a, A_b をそれぞれエージェント a, b の論証とする．

1. $Sentence(A_a) \equiv \neg Sentence(A_b)$ かつ A_b の優先度が A_a の優先度よりも低くなければ，(A_a, A_b) は a から b への rebut という．
2. $\neg Sentence(A_a) \in Grounds(A_b)$ かつ A_b の優先度が A_a の優先度よりも低くなければ，(A_a, A_b) は a から b への undercut という．
3. a から b への *rebut* または *undercut* を a から b への攻撃という．

以後は提案者 P が最初に提案 φ を発言し，反論者 C がそれに反論することで議論が始まるものとする．議論者 P,C の知識ベースをそれぞれ $\mathbf{K}_\mathrm{P}$, $\mathbf{K}_\mathrm{C}$ で表す．

Definition 5 (議論フレームワーク) $AR_{\mathbf{K}_\mathrm{P}}$, $AR_{\mathbf{K}_\mathrm{C}}$ をそれぞれ優先度の付加された P, C の可能な論理式の集合とする．また，$AT_{\mathbf{K}_\mathrm{P} \rightarrow \mathbf{K}_\mathrm{C}}$, $AT_{\mathbf{K}_\mathrm{C} \rightarrow \mathbf{K}_\mathrm{P}}$ をそれぞれ P から C, C から P への攻撃とする．P,C 間の議論フレームワーク $AF(\mathbf{K}_\mathrm{P}, \mathbf{K}_\mathrm{C}, \nu)$ は *4* 項組 $\langle AR_{\mathbf{K}_\mathrm{P}}, AR_{\mathbf{K}_\mathrm{C}}, AT_{\mathbf{K}_\mathrm{P} \rightarrow \mathbf{K}_\mathrm{C}}, AT_{\mathbf{K}_\mathrm{C} \rightarrow \mathbf{K}_\mathrm{P}} \rangle$ で定義される．

2.2 議論木

Definition 6 (提議) 発話者であるエージェント a と論証 $A \in AR_{\mathbf{K}_\mathrm{a}}$ の組を a の提議という．提議 $M = (a, A)$ における a, A をそれぞれ $Ply(M)$, $Arg(M)$ と表す．

Definition 7 (議論筋) 議論フレームワーク $AF(\mathbf{K}_\mathrm{P}, \mathbf{K}_\mathrm{C}, \nu)$ における提案 φ の議論筋は以下の条件を満たす空でない提議の有限列 $[M_1, \ldots, M_n]$ $(i = 1, \ldots, n)$ として定義される．

1. $Ply(M_1) = P$, ただし $Sentence(Arg(M_1)) = \varphi$.
2. i が奇数ならば，$Ply(M_i) = P$, i が偶数ならば，$Ply(M_i) = C$.
3. すべての i $(1 \leq i \leq n-1)$ に対して M_{i+1} は M_i に対する攻撃である．
4. $Arg(M_n)$ に対する攻撃はない．
5. 任意のペア i, j $(1 \leq i \neq j \leq n)$ に対して $M_i \neq M_j$.
6. $S(Arg(M_1)) \cup S(Arg(M_3)) \cup S(Arg(M_5)) \cup \ldots \cup S(Arg(M_o))$ および $S(Arg(M_2)) \cup S(Arg(M_4)) \cup S(Arg(M_6)) \cup \ldots \cup S(Arg(M_e))$ はいずれも無矛盾である．ただし o および e は

それぞれ n を越えない最大の奇数，偶数である．D 上に出現する P,C の論証の集合 $S(Arg(M_1)) \cup S(Arg(M_3)) \cup S(Arg(M_5)) \cup \ldots \cup S(Arg(M_o))$ および $S(Arg(M_2)) \cup S(Arg(M_4)) \cup S(Arg(M_6)) \cup \ldots \cup S(Arg(M_e))$ をそれぞれ $S_P(D)$, $S_C(D)$ と表す．

議論筋 D の最後の発話が P の提議ならば P は D で勝つといい，そうでなければ P は D で負けるという．

Definition 8 (議論木) $AF(\mathbf{K}_\mathrm{P}, \mathbf{K}_\mathrm{C}, \nu)$ における提案 φ の議論木は深さ *0* である根ノードが空で，深さ *1* から始まるすべての枝が $AF(\mathbf{K}_\mathrm{P}, \mathbf{K}_\mathrm{C}, \nu)$ における φ の議論筋になっているもの[†1]である．

Definition 9 (候補部分木) 議論木 T の候補部分木は，T における C の提議に対応するノードに対しては *1* つの子ノードを，P の提議に対応するノードに対してはすべての子ノードをそれぞれ含むような部分木である．

Definition 10 (解部分木) 解部分木はそのすべての議論筋で P が勝つような候補部分木である．

候補部分木は，P に複数の提議が可能な場合に，P のとる選択に対応するものである．解部分木は， C によるすべての反論を論破できるように P が戦略をとった場合に相当する．図 1 において，(a) は議論木の例，(b) および (c) はその候補部分木を示す．(b) は解部分木になっている．

一般に，議論木の勝敗は T の解部分木の有無で判定される．すなわち，解部分木が存在すれば P の勝利であり，そうでなければ P の敗北である．本研究では，このように定義される勝敗を「静的勝敗」と呼び，後に導入する「動的勝敗」と区別する．

3 知識ベースの変化を伴う議論システム

通常の議論システムは議論木の単一の議論筋における議論を独立に扱って議論の勝敗を定義していた．本研究では，1 つの議論筋の議論が終了したあとも別の議論筋の議論が継続するようなシステムを考える．その場合，複数の議論筋で行われた議論が互いに影響をおよぼす．

まず，各エージェント a に対し，履歴 $\mathbf{H}_a$ を a が勝利した議論筋のすべての a の提議に含まれる式の集合として定義する．勝利した議論筋に関しては，エージェントはそれまでの自分の発言に責任をもつという考えから $\mathbf{H}_a$ は無矛盾でなければならない．しかし，敗北した場合は，別の面からの反論を試みるため，前に言ったことと矛盾したことを言ってもよい．

動的議論筋は一般の議論筋に履歴の概念を加えて拡張することで定義される．

Definition 11 (動的議論筋) $AF(\mathbf{K}_\mathrm{P}, \mathbf{K}_\mathrm{C}, \nu)$ における φ, $\mathbf{H}_P$, $\mathbf{H}_C$ に対する動的議論筋 $D = [M_1, \ldots, M_n]$ は定義 *7* で示した一般の議論筋の定義に以下の *1* 条件を加えたものである．

7. 各 i *(*$1 \leq i \leq n$*)*.$\mathbf{H}_{Ply(M_i)} \cup S(Arg(M_i))$ は無矛盾である．

動的議論筋 D, 提議 M は議論木における枝，ノードにそれぞれ対応するため，以後 D, M をそれぞれ枝，ノードと呼ぶことがある．

本研究では議論の実行を，議論木の 1 つの枝を選択し，commitment store および両エージェントの履歴を更新して議論木を変更していくものと考える．commitment store はそれまでの議論で提出されたすべての議論に含まれる式の集合である．エージェントはそれぞれの知識ベースと commitment store に属する式を使って議論する．

議論はまず，初期議論木の枝を選択することで開始される．議論はその枝に沿って進行し，葉ノードに至ればその枝は停止する．停止した時点で commitment store が更新される．議論の進行に伴い新しいノードが加わる場合がある．続いて，木の再構築がなされた後，別の枝が選択される．停止した枝は，他の枝の実行によってその枝の葉ノードに未マークのノードが付加された場合再開することができる．枝の選択においては，発話順を維持する必要がある．すなわち、別の枝に移る時は，停止したノードで発話していたエージェントの相手の発話から始まる枝しか選択できない．

知識ベースの更新に伴って新しくノードが生成される場合がある．この状況を説明するために脅威の概念

†1 本研究では対象となるノードから葉ノードまでの経路を枝と呼ぶ．

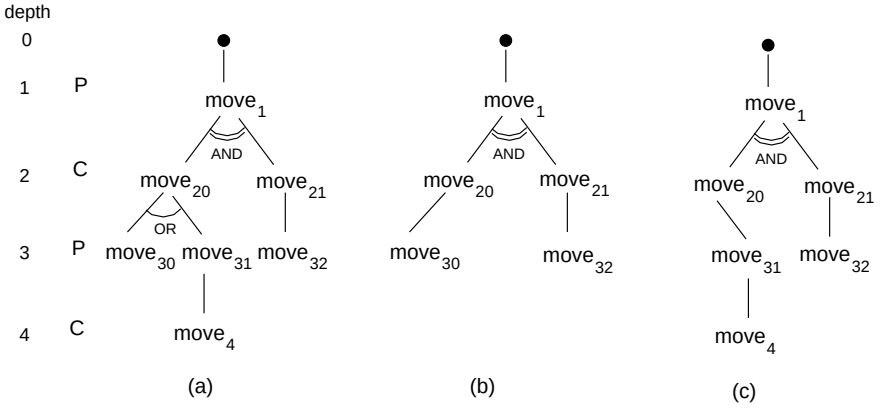


図 1　議論木とその候補部分木

を導入する．

Definition 12 (脅威) M, M' を $AF(\mathbf{K}_{\mathrm{P}}, \mathbf{K}_{\mathrm{C}}, \nu)$ における議論木 T における提議とする．$S(Arg(M))$ が M' を攻撃する提議を *1* 個以上新たに生成する場合，M は M' の脅威という．このとき，M, M' をそれぞれ脅威元, 脅威先という．

直観的には脅威は相手に有利な情報をもたらすような提議である．ある提議は同一枝の自分の提議に対する脅威になる場合もある．

以下に議論木の実行の形式的な定義を示す．

Definition 13 (実行可能なノード) 枝 $D = [M_1, \ldots, M_n]$ に対してその枝を選択する直前に実行したノードの発話エージェントを t とする．$M_1, \ldots, M_{i-1}$ *(*$1 \leq i \leq n$*)* がマーク済み，$M_i, \ldots, M_n$ が未マーク，$Ply(M_i) = t$ ならば，ノード M_i は実行可能という．

Definition 14 (枝の実行) 枝 $D = [M_1, \ldots, M_n]$, 履歴 $\mathbf{H}_P, \mathbf{H}_C$ *commitment store* $\mathbf{K}$ に対し D の i *(*$1 \leq i \leq n$*)* からの実行は以下のように定義される．ただし，t をその枝を選択する直前に実行したノードの発話エージェントとする．

1. $M_i, \ldots, M_n$ をマークする．
2. $\mathbf{K} = \mathbf{K} \cup \bigcup_{k=i}^{n} S(Arg(M_i))$ とおく．
3. **if** $Ply(M_n) = P$,
   then $t = C$ とし，$\mathbf{H}_P = \mathbf{H}_P \cup S_P(D)$ とする．
   if $Ply(M_n) = C$,
   then $t = P$ とし，$\mathbf{H}_C = \mathbf{H}_C \cup S_C(D)$ とする．

Definition 15 (停止/再開) ある枝のすべてのノードの実行が終了すれば，D は停止されたという．停止された枝 D に対して，新しく実行可能なノードがその枝の葉ノードに加わり，再び D が選択された場合，D は再開されたという．

知識ベースの変更を伴う議論手続き (*APKC2*)

$AF(\mathbf{K}_{\mathrm{P}}, \mathbf{K}_{\mathrm{C}}, \nu)$ を議論フレームワーク，φ を提案された式とする．

[STEP 1(初期化)]

$\mathbf{K} = \emptyset$, $\mathbf{H}_P = \emptyset$, $\mathbf{H}_C = \emptyset$, $turn = P$ とする．$AF(\mathbf{K}_{\mathrm{P}}, \mathbf{K}_{\mathrm{C}}, \nu)$ における φ, $\mathbf{H}_P, \mathbf{H}_C$ に対する初期議論木をすべてのノードを未マークとして作成する．

[STEP 2(議論の実行)]

if どの枝にも実行可能なノードがないならば,
  if $turn$=P, **then** P の敗北として終了．
  else $turn$=C, **then** P の勝利として終了．
else 枝を選択して実行可能なノードから実行する．

[STEP 3(木の再構築)]

$AF(\mathbf{K}_{\mathrm{P}} \cup \mathbf{K}, \mathbf{K}_{\mathrm{C}} \cup \mathbf{K}, \nu)$ のおける φ, $\mathbf{H}_P, \mathbf{H}_C$ に対する議論木を再構築する．その際，ノード N と M が同一で，N はマークされ M が未マークの場合，M をマークする．

go to STEP 2.

APKC2 の STEP 3 で新たなノードが付加される

と，候補部分木が複数に分かれる場合がある．

Definition 16 (継続候補部分木) 候補部分木 CT に対してノードが付加されることによって 2 つ以上の候補部分木が生成される場合，これらの部分木を CT の継続候補部分木と呼ぶ．

また，候補部分木 CT において葉ノードが C の提議であるような別の候補部分木の枝 D を実行する場合，それは CT から連続するものと判断し，D を CT につなげた木を候補部分木と考える．

$\mathbf{K}$ の任意の要素は $\mathbf{K}_{\mathrm{P}}$ または $\mathbf{K}_{\mathrm{C}}$ に含まれる．$\mathbf{K}_{\mathrm{P}}$ も $\mathbf{K}_{\mathrm{C}}$ も有限集合なので，新たに生成されるノードの数は有限個である．従って，*APKC2* は必ず停止する．

APKC2 の実行においては，エージェントが順に提議を出し，出せなくなった方が負ける．一連の実行は，1 つの議論パターンを実行シミュレートしているにすぎない．両者とも選択できる枝は複数あるので，どの枝から実行するかによって最終的に得られる議論木は異なり，勝敗も変化する．

例 1

図 2 の議論木を考える．M_3 と M_5 はそれぞれ脅威元，脅威先の関係，M_6 は脅威によって新たに生成されたノードとする．左の枝から実行した場合，*APKC2* はノード M_1, M_2, M_3 の順に実行した後，木の再構築によって M_6 が付加される．続いて M_4, M_5, M_6 の順に実行し，P の敗北で終了する．一方，右の枝から実行した場合，*APKC2* はノードを M_1, M_4, M_5 の順に実行した後停止する．続いて M_2, M_3 の順に実行した後木の再構築によって M_6 が付加され，右の枝が再開され，M_6 を実行し P の敗北で終了する．

例 2

図 3 の議論木を考える．M_2 と M_4 はそれぞれ脅威元，脅威先の関係，M_5 は脅威によって新たに生成されたノードとする．左の枝から実行した場合，*APKC2* はノード M_1, M_2, M_3 の順に実行した後，木の再構築によって M_5 が付加される．続いて M_4, M_5 の順に実行し，P の勝利で終了する．一方，右の枝から実行した場合，*APKC2* はノードを M_1, M_4 の順に実行した後停止する．次は P の番だが，P の提議から開始して P の提議で終了するような枝がないので実行はこれ以上すすまず P の敗北で終了する．

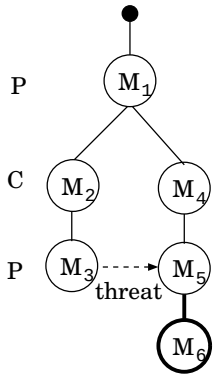


図 2　脅威のある議論木の実行例

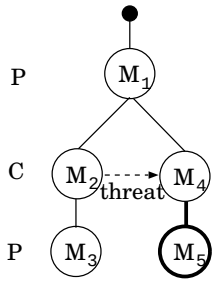


図 3　枝の実行順序に影響される例

例 3

図 4 のような議論木を考える．左の枝から実行した場合，*APKC2* はノードを M_1, M_2 の順に実行した後，木の再構築によって M_4 が付加される．続いて M_3, M_4 の順に実行し，P の敗北で終了する．一方，右の枝から実行した場合，*APKC2* はノード M_3 を実行した後停止する．次は C の番だが，C の提議から開始して C の提議で終了するような枝がないので実行はこれ以上すすまず P の勝利で終了する．

以下では，*APKC2* に基づいて動的勝敗を定義する．

Definition 17 (動的解部分木) CT を初期議論木の候補部分木とする．CT の実行順序にかかわりなく APKC2 が P の勝利となって終了するか，CT が P の勝利となって終了する継続部分木をもつかのいずれかであるとき，CT を 動的解部分木という．

Definition 18 (議論木の動的勝利) 議論木が動的解部分木をもつならば，P はその議論木に動的勝利し，

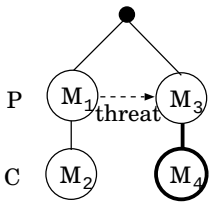


図 4　候補部分木にまたがる実行

そうでなければ，P はその議論木に動的敗北する．

P の動的勝利とは，相手がどのような順で反論してきても必ず勝てるような選択方法があることを示している．

4　動的勝敗の判定

APKC2 は議論手続きに対する実行モデルを与える．すなわち，実際の議論の進行をシミュレートすることができる．しかし，もし議論の結果の勝敗だけを問題にするのならば，もっと簡単なアルゴリズムが存在する．

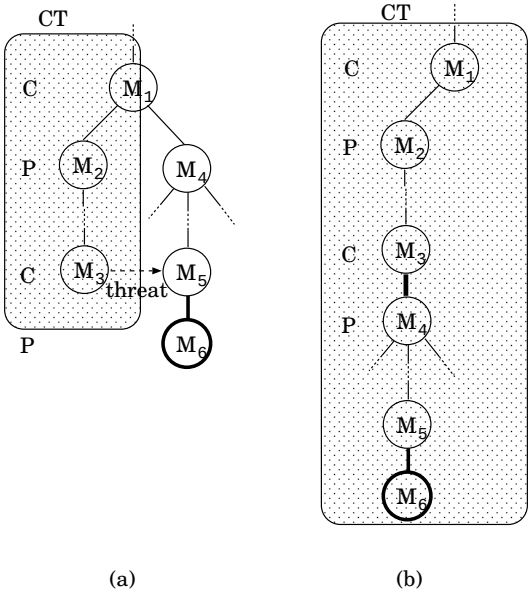
Definition 19 (無矛盾候補部分木) CT を候補部分木とする．$Ply(M) = Ply(M') = P$, $\psi \in S(Arg(M))$, $\neg\psi \in S(Arg(M'))$ を満たすような提議 M, M' と式 ψ が存在しないならば，CT は無矛盾候補部分木という．

候補部分木の挿し木 (図 5)

議論木を T, その候補部分木を CT とし，CT 内に脅威元，CT 外に脅威先がある場合，脅威元，脅威先の提議をそれぞれ M_r, M_d とする．$Ply(M_r) = Ply(M_d) = C$ ならば，M_r, M_d の共通の祖先ノードで根ノードからもっとも遠い M に対して以下の操作を行う．

M_d の祖先ノードで M を親ノードとしてもつノードの提議を M' とする．M_r を親ノード，M' を子ノードとして枝をつなぎ，M' を根ノードとする部分木を M の下につないで新たに生成されたものを候補部分木とする．この操作を**候補部分木の挿し木**と呼ぶ．

図 6 に議論木の動的判定を行うアルゴリズム *JC2* を示す．本アルゴリズムは *APKC2* と同様の理由で



(a)　(b)

図 5　候補部分木の挿し木

必ず停止する．

APKC2 と *JC* の直観的な違いは，*APKC2* が枝単位で実行を逐次的にシミュレートしているのに対して，*JC2* は候補部分木単位の実行をバッチ処理している点である．勝敗の判定という観点からは，*APKC2* よりも *JC2* の方はコスト的に有利である．

先に述べた 3 つの例において *JC2* における判定と *APKC2* における動的勝敗の判定が一致するかどうかを調べる．

例 1 については，*APKC2* では動的解部分木が存在しないので，P の動的敗北になる．*JC2* でも P の敗北と判定される．

例 2 については，*APKC2* では動的解部分木が存在しないので，P の動的敗北になる．しかし，*JC2* では，最終的に得られる木の葉ノードがすべて P になるため，P の勝利と判定される．この場合は両者は一致しない．

例 3 については，候補部分木が 2 つ含まれ，初期状態では右側の候補部分木は M_3 しかノードがない．*APKC2* では 右の候補部分木が動的解部分木になっているため，P の動的勝利になる．*JC2* においては，左の候補部分木を選択した場合は，M_2 の下に挿し木

議論木の勝敗の判定 (*JC2*)

$AF(\mathbf{K}_{\mathrm{P}}, \mathbf{K}_{\mathrm{C}}, \nu)$ に対する議論木を T とし，これに含まれるすべての候補部分木を未検査とする．
[STEP 1]
if T に未検査で無矛盾な候補部分木がなければ，
　then P の動的敗北としてし終了する．
else 未検査で無矛盾な候補部分木の 1 つを CT とおく．
[STEP 2]
$\mathcal{K} = \bigcup_{D \in CT} S_P(D) \cup \bigcup_{D \in CT} S_C(D)$ とおき，
$AF(\mathbf{K}_{\mathrm{P}} \cup \mathcal{K}, \mathbf{K}_{\mathrm{C}} \cup \mathcal{K}, \nu)$ に対する議論木を再構築してそれを T とおく．また，このとき CT の葉ノードに新たにノードを付加することで拡張された CT を CT' とおく．
[STEP 3]
if CT' が CT と一致するならば，
　then $CT' := CT$ として goto STEP2.
else goto STEP4.
[STEP 4]
if CT に含まれる候補部分木で未検査で無矛盾なものがなければ，
　then CT を検査済みとして goto STEP1.
else そのような候補部分木の 1 つを選択し CT'' とする．
[STEP 5]
if CT'' の葉ノードがすべて P の提議ならば，
　then P の動的勝利として終了する．
else goto STEP6.
[STEP 6]
T を調べる．
if CT'' に属する葉ノードで C の提議になっているものが脅威元，CT'' に属さないノードが脅威先になっているものがあれば，
　then CT'' に挿し木をした候補部分木を新たに CT とおいて goto STEP2.
else CT'' を検査済みとして goto STEP4.

図 6　議論木の勝敗判定アルゴリズム

よって M_3, M_4 がつながり，葉ノード M_4 が C の提議になっているため，P の敗北になるが，右の候補部分木を選択した場合は明らかに P の勝利である．すなわち，*JC2* においても P の勝利と判定される．

5　おわりに

本研究の特徴はこれまでの議論システムが議論木の単一枝を独立して勝敗の判定をしていたのに対して，議論を実行という動的な観点からとらえて複数の枝を継続的に実行されるものとして扱っている点であ

る．このような観点をとるために，提議同士の脅威関係という概念を導入した．本発表では，議論の勝敗を判定するアルゴリズムを候補部分木にまたがる脅威がある場合についても扱えるように拡張し，このアルゴリズムと議論をシミュレーションした結果の一致について論じた．ここであげた反例以外では両者が一致すると予想されるが，厳密な証明はまだであり，さらに，一致するための条件を見つけることも今後の課題である．

参 考 文 献

[1] L.Amgoud, S.Parsons, and N.Maudet: Arguments, dialogue, and negotiation, ECAI2000, pp.338-342, 2000.

[2] L.Amgoud, Y.Dimopolos and P.Moraitis: A general framework for argumentation-based negotiation, ArgMAS2007, pp.1-17, 2007.

[3] L.Amgoud and S.Vesic: Repairing preference-based argumentation frameworks, IJCAI2009, pp.665-670, 2009.

[4] T.Bench-Capon and P.Dunne: Argumentation in artificial intelligence, Artificial Intelligence, 171, pp.619-641, 2007.

[5] C.Cayrol, F.D.de St-Cyr, and M-C Lagasquie-Shiex: Revision of an argumentation system. pp.124-134, KR2008, 2008.

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

[7] M.Falappa, G.Kern-Isberner and G.R.Simari: Explanations, belief revision and defeasible reasoning, Artificial Intelligence, 141(1-2), pp.1-28, 2002.

[8] A.García, and G.Simari: Defeasible logic programming: an argumentative approach. Theory and practice of logic programming, 4(1), pp.95-138, 2004.

[9] A.García, C.Chesñevar, N.Rotstein, and G.Simari: An abstract presentation of dialectical explanations in defeasible argumentation, ArgNMR07, pp.17-32, 2007.

[10] S.Kraus, K.Sycara and A.Evenchik: Reaching agreements through argumentation: a logical model and implementation, Artificial Intelligence, 104(1-2), pp.1-69, 1998.

[11] S.Modgil: Reasoning about preferences in argumentation frameworks, Artificial Intelligence, 173(9-10), pp.901-1040, 2009.

[12] M.O.Moguillansky, et al.: Argument theory change applied to defeasible logic programming, AAAI2008, pp.132-137, 2008.

[13] H.Prakken: Combining skeptical epistemic reasoning with credulous practical reasoning. COMMA 2006, pp.311-322, 2006.

[14] F.Paglieri and C.Castelfranchi: Revising beliefs through arguments: bridging the gap between argumentation and belief revision in MAS, ArgMAS2004, pp.78-94, 2004.

[15] K.Okuno and K.Takahashi: Argumentation system with changes of an agent's knowledge base, IJCAI2009, pp.226-232, 2009.

[16] K.Okuno and K.Takahashi: Argumentation system allowing suspend/resume of an argumentation line,. ArgMAS2010, pp.145-162, May, 2010.

[17] I.Rahwan, and G.Simari (eds.): *Argumentation in Artificial Intelligence*, Springer, 2009.