好みに基づく分散 ATMS を用いたグループスケジュール管理システムについて

On a group schedule management system using preference-based distributed ATMS

伊藤 孝行 † 新谷 虎松 †
Takayuki Ito † Toramatsu Shintani †
† 名古屋工業大学知能情報システム学科
† Nagoya Institute of Technology, Dept. of Intelligence and Computer Science

Abstract: In this paper, we propose a group schedule management system using a preference-based distributed ATMS (Assumption-based Truth Maintenance System). We can classify group schedules into three types as follows: local schedules, local & share schedules, and global schedules. The problem is that the existing distributed ATMS cannot reflect users' preferences. Therefore, we introduce importance of schedules in order to reflect users' preferences into the schedules. The advantages of our system can be shown as follows:(1)Our system can reflect users' preferences by introducing importance of schedules. (2)By using preference-based distributed ATMS, we can maintain consistency of the local schedules, local & share schedules, and global schedules. (3)Because agents negotiate autonomously in our system, we can preserve the users' privacy.

1 はじめに

CSCW やグループウェアなどの分野においてネットワークを介してグループの活動の支援をするシステムに関する研究が盛んに行われている [8][18]．さらにグループの活動の支援のためにエージェント [12] などの知的な情報処理技術を導入することによる効果的な支援が期待されている [10][11]．本論文でエージェントとは，ネットワーク上において人間の代理として自律的かつ協調的に行動するソフトウェアを指す．エージェントが支援すべきグループの活動の一つにグループのスケジュール管理 [9] がある．

グループにおいて，公的なスケジュールと個人的なスケジュールの間の整合性を保ちながら作成するのは困難な作業である．なぜなら，グループの公的なスケジュールを生成する時には，以下の３点を考慮する必要があるからである．(1) 個々のメンバーのスケジュール間の整合性の管理．(2) 個々のメンバーのスケジュールに対する好みの反映．(3) メンバーの個人的なスケジュールのプライバシーの保護の面からの維持．

以上の３点を考慮した上で，本論文では現実的なグループのスケジュールのために，グループのスケジュールをスケジュールの特徴に基づいて３つのタイプに分類する．そして整合性管理機構である ATMS[3] に基づいて，(1) 個々のメンバーのスケジュール間の整合性を管理する．個々のメンバーのスケジュールに対する重要度を導入することにって，(2) メンバーの個人的なスケジュールのプライバシーを保護しながら，(3) 個々のメンバーのスケジュールに対する好みの反映が可能な分散整合性維持手法を提案する．

文献 [17] では，グループ全体の会議などのある一つのイベントをどのような日時にスケジュールするかを決定するための手法として，エージェント間の説得に基づく交渉による手法を提案した．文献 [17] では，あるイベントがグループのスケジュールとして決定した時，決定されたイベントに競合したスケジュールを持っているユーザは，決定されたイベントと一貫性を維持するためにスケジュールをやり直す必要があった．本論文で

†連絡先:〒 466-8555 名古屋市昭和区御器所町
名古屋工業大学知能情報システム学科新谷研究室
TEL:(052)735-5471
FAX:(052)735-5477
E-mail: itota@ics.nitech.ac.jp

は，一貫性の維持を含めたグループのスケジュールの管理手法を提案する．

本論文では，ユーザ個人の好みを反映しながらスケジュールの整合性を管理するスケジュール管理システムを提案する．本論文の構成は，２章で既存の分散 ATMS の説明と本グループスケジューリング管理システムの構成を示し，現実的なグループのスケジュールを基にしてグループのスケジュールを３つのタイプに分類する．３章では，３つのタイプのスケジュールをユーザの好みを反映しながら管理する手法を示す．４章では，関連研究と本論文の手法の相違を明らかにすることによって，本論文の手法の特長を示し，５章はまとめとしての結論である．

2 グループスケジュール管理システム

2.1 基本的な分散 ATMS

本節では基本的な分散 ATMS について簡単に説明する．まず ATMS について説明する．ATMS では，データの基本単位をノードと呼び，

$$< datum, label, justifications >$$

で表される．datum はデータの内容，label は仮説のリスト（環境と呼ばれる）の集合，および，justifications はそのデータの依存関係を表す．依存関係は，前提部と結論部を持つ．ノードの種類には，前提ノード，仮説ノード，仮定されたノード，および導出されたノードがある．前提ノードは常に真であり，支持する仮説や依存関係を持たない．前提 p に対する前提ノードは $< p, \{\{\}\}, \{()\} >$と表される．仮説ノードは真と仮定されたデータを表し，それ自身からなる単一の環境をラベルとして持つ．例えば，仮説ノード $< A, \{\{A\}\}, \{(A)\} >$ は，仮説 A を表す．仮定されたノードは，前提でも仮説でもなく，ある仮説による依存関係を持つ．例えば，仮説 A に依存関係のある仮定されたデータ a は，仮定されたノード $< a, \{\{A\}\}, \{(A)\} >$として表される．その他のノードを導出されたノードと呼ぶ．例えば，$< w = 1, \{\{A, B\}, \{C\}\}, \{(b), (c, d)\} >$は，“$w = 1$ がノード b または，ノード c とノード d から導出され，環境 $\{A, B\}$ または $\{C\}$ で成立する，” ということを表している．また，矛盾を生じるような仮説の組は nogood データベースとして保持される．

新たな依存関係が与えられると ATMS は依存関係の結論部をデータとしてもつノードを生成する．依存関係の結論部が矛盾（$\perp$）であるときは，その前提部を nogood 環境として nogood データベースに登録する．ノードの label は，ノードが表すデータを結論に持つ各依存関係の前提部の環境の直積を求めることによって得る．求められた label の中の環境のうち，nogood 環境の一つを包摂する環境は，矛盾するので取り除かれる．

一般的に仮説の集合 A と依存関係の集合 Jにより導かれるすべての矛盾しないノード n の集合をコンテキスト Cと呼び，$A \cup J \vdash n,\ n \in C,$かつ$\perp \notin C$($\perp$ は矛盾を表す) と定義される．

分散 ATMS は複数のエージェントが ATMS を個々に持ち，互いに交渉することによってマルチエージェント環境において共有データの整合性を管理するシステムである．一つの ATMS を各エージェントが共有する集中管理型の整合性維持も考えられるが，以下のような短所がある．（１）情報が一括管理されるために，ユーザ個々のスケジュールなどの情報が一括管理されてしまう．これはプライバシーの面から好ましくない．（２）集中管理型の整合性維持システムは，あるエージェントが整合性維持システムを用いている時には他のエージェントが使えないなどのボトルネックが発生することがある．

2.2 スケジュールの好みの反映

本システムでは，ある一日に行われる予定の一つのイベントを一つの仮説とする．イベントには例えば，会議などがある．つまり本システムでは，ある会議 m を 1998 年 10 月 24 日か 1998 年 10 月 25 日のどちらかで行うという表現を,「会議 m を 1998 年 10 月 24 日に行う」,「会議 m を 1998 年 10 月 25 日に行う」という２つの仮説として表現する．あるイベントに付随するイベントはイベント間の依存関係を用いて表現する．イベントに付随するイベントには例えば，会議とその会議の準備などが挙げられる．後件にあらわれるスケジュールが実行されてしまったら前件のスケジュールはすべて前提（常に真）となる．

ユーザは仮説各々について重要度を添付する．重要度は，仮説が成立することに対するユーザの願望の度合を表している．仮説の重要度は，９以下の整数値で表す．数値が大きければ大きいほど，その仮説は重要であることを示す．１，３，５，７，および９はそれぞれ，“やや重要”，“重要”，“かなり重要”，“非常に重要”，および “極めて重要” を示す（２，４，６，および８は中間の値として使われる）．例えば，あるユーザが「会議 m を 1998 年 10 月 25 日に行う」という仮説に重要度 7 を添付した場合，そのユーザは「会議 m を 1998 年 10 月 25 日に行うことが非常に重要である」という好みを

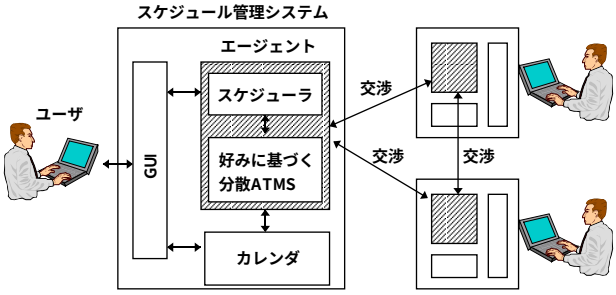

図 1: システム構成

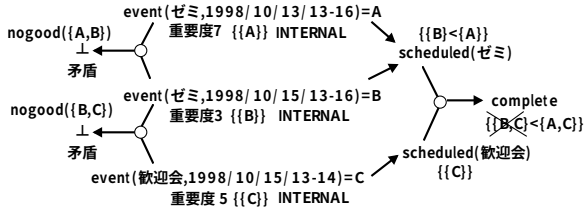

図 2: Local な整合性維持の例

持つことが分かる．

スケジュールにおける矛盾は，同じイベントが違う日付に行われるとき，異なる会議が同じ日付に行われるときに発生するものとする．例えば「会議 m を 1998 年 10 月 24 日に行う」,「会議 m を 1998 年 10 月 25 日に行う」という 2 つの仮説は矛盾することになる．

本システムの構成を図 1で示す．各ユーザはカレンダとエージェントを持つ．エージェントは，ユーザのスケジュールを管理するためのスケジューラとスケジュールの整合性の維持のために本システムを持つ．エージェントは，スケジュールの整合性を維持するために他のエージェントと交渉を行う．ユーザは，GUI を用いてエージェントおよびカレンダとインタラクションする．

2.3 スケジュールのタイプ

現実的なグループのスケジュールを考えた場合，スケジュールが影響する範囲によってスケジュールを分類することができる．そこで本システムではグループにおけるスケジュールを以下の 3 つのタイプに分類する．(1)Local スケジュール，(2)Local& Share スケジュール，および (3)Global スケジュールである．(1)Local スケジュールは，グループ内における個人的なスケジュールで，グループ内の他のメンバーとは共有しないスケジュールである．(2)Local& Share スケジュールは，グループ内における個々のメンバー同士が共有するスケジュールである．(3)Global スケジュールは，グループ内におけるメンバー全員が共有するスケジュールである．Global なスケジュールには，トップダウンに決定されるスケジュールとボトムアップに決定されるスケジュールが考えられる．例えば，大学の研究室を一つのグループとすると，研究室内の友人との食事会は Local&Share スケジュールである．個人的なアルバイトは Local スケジュールである．大学の卒業式はトップダウンに決定される Global スケジュールであり，研究室内の学生が提案する研究室旅行はボトムアップに決定される Global スケジュールである．

3 分散スケジュール管理プロセス

3.1 Local スケジュールの管理

Local スケジュールの整合性の維持について述べる (図 2)．Local な整合性の維持の方法は基本的な ATMS に基づいている．本システムでは，日付，時間，および名前からなる一つのイベントを一つの仮説とする．例えば，図 2の $event(\text{ゼミ}, 1998/10/13/13-16) = A$ は，1998 年 10 月 13 日の 13〜16 時のゼミというイベントで，A という仮説であることを表す．重要度は 7 が添付されている．本システムは，各イベント間の依存関係を考慮した上で矛盾のないスケジューリングをする．エージェントは自分の仮説は $INTERNAL$ とし，他から受信した仮説は $EXTERNAL$ とする．

本システムでは，各ノードのラベルにおける各環境の重要度を計算し，重要度が大きい順に列べる．ラベルとは，そのノードが成り立つ環境を表すものである．図 2に示す通り，ノード $scheduled(\text{ゼミ})$ のラベルは $\{\{B\} < \{A\}\}$ となっており，ノード $scheduled(\text{ゼミ})$ が環境 $\{B\}$ と環境 $\{A\}$ で成り立つことを表している．つまり仮説 A または仮説 Bの基でゼミがスケジューリングでき，かつ A の方が好ましいことを表している．環境の重要度は，その環境に含まれる仮説の重要度の中の最大値とする．各ユーザのスケジュールは図 2に示す通り，$complete$ ノードのラベルの中で最も好ましい環境に含まれる仮説集合とする．$complete$ ノードのラベルが空でないとき，スケジュールが少なくとも一つ存在するので，スケジューリングが成功したという．

3.2 Local&Share スケジュールの管理

Local&Share スケジュールの整合性を維持する方法を示す．Local&Share スケジュールの流れは以下の通りである．(1) あるエージェントが重要度を添付した仮説

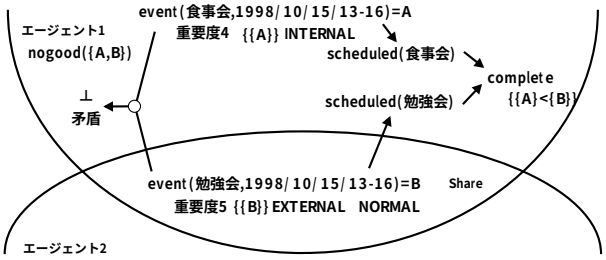

図 3: Local&Share な整合性維持の例 (1)

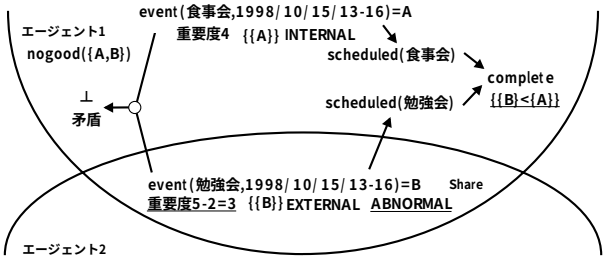

図 4: Local&Share な整合性維持の例 (2)

を他のエージェントに送信する．(2) 受信したエージェントは，受信した仮説が自分の他の仮説と矛盾しなければ受け取る．もし受信した仮説を含む環境が，$complete$ ノードのラベル中で最も好ましいなら，受信した仮説の重要度の調整を試みる．(2) では重要度の調整は，自分のスケジュールをなるべく優先するというエージェントの利己主義に基づいた調整である．一般にグループのメンバーには，おおげさに重要度を提示するメンバーが存在することがある．そこで本システムでは，Local&Share スケジュールを調整する場合に，あるエージェントが提示する重要度を他のエージェントが評価するための指標として，エージェントの信頼度を導入する．仮説に極端な重要度を続けて添付するエージェントの信頼度は低くなる．信頼度が極端に低いエージェントが送信する $EXTERNAL$ な仮説は，$ABNORMAL$ な仮説とし，それ以外の $EXTERNAL$ な仮説は $NORMAL$ とする．エージェントは $ABNORMAL$ な仮説を受信した時は，その仮説の重要度を (重要度) -1〜(重要度) -2 の範囲で調整する．具体的には自分の他の仮説が $complete$ ノードにおいて最も優先されるように受信した仮説の重要度を下げる．

図 3と図 4にエージェント 1 とエージェント 2 間の Local&Share なスケジュールの整合性維持の例を示す．図 3の例 (1) では，エージェント 2 が $event$(勉強会, $1998/10/15/13-16$) という $EXTERNAL$ かつ $NORMAL$ な仮説を受理したとする．この場合，エージェント 1 は重要度をそのまま 5 として受け入れる．結果として $complete$ ノードのラベルから，仮説 Bが優先される．一方，図 4の例 (2) では仮説 $event$(勉強会, $1998/10/15/13-16$) が，$ABNORMAL$ となっておりエージェント 2 の信頼度は低い．そこでエージェント 1 は重要度 5 を信頼度に基づき調整し $5-2=3$ とした結果，重要度 3 として受け入れる．結果として $complete$ ノードのラベルから，仮説 A が優先される．

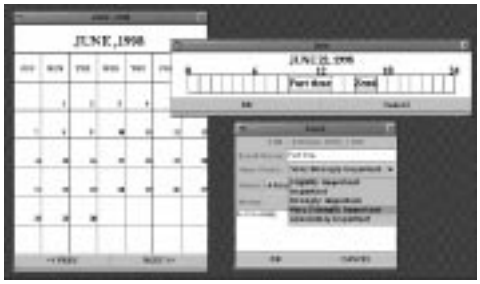

図 5: ユーザインターフェース

3.3 Global スケジュールの管理

本節では，Global スケジュールの整合性を維持する方法を示す．Global スケジュールのトップダウンなスケジューリングは以下の通りである．(1) マネージャとなるエージェントがイベントを他のエージェントに送信する．(2) 受信したエージェントは重要度 9 の最も重要な仮説として受け取る．送信されたイベントは重要度が 9 であるから，どのエージェントも最も重要とする．

Global スケジュールのボトムアップなスケジューリングは以下の通りである．(1) グループの中の一つのエージェントが，決定すべきイベントを他のエージェントに送信する．(2) 提案したエージェントとその他のエージェントが Local&Share なイベントとして，Local&Share なスケジューリングを行う．(3) すべてのエージェントが受け入れ可能になればスケジューリングは成功とする．

4 Aglets を用いた実装

本システムはモバイルエージェントに基づいて実装されている．モバイルエージェントは，IBM の ASDK(Aglets Software Development Kit)[14] を利用して構築されている．ASDK はモバイルエージェントを Java 言語 [1] 用いて構築するためのパッケージである．本システムはすべて Java に基づいて実装されているため，Java が利用可能な環境では実行可能である．ASDK

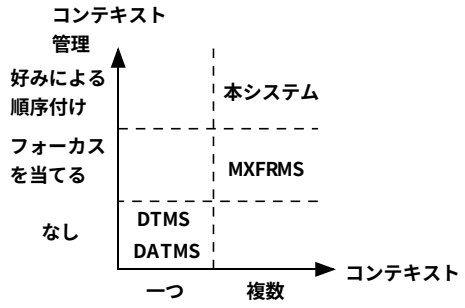

図 6: コンテキストの数と管理手法について

ではモバイルエージェントを Aglet というインターネット上におかれたホストからホストへ移動することのできる Java オブジェクトとして構築する．ASDK を用いることによって，セキュリティに対して頑健なエージェントを容易に構築することが可能となる．本システムのユーザインターフェースを図 5に示す．

5 議論

本章では関連研究と本研究を比較することによって，本システムの特長を明確にする．

文献 [4] では仮説に好みではなく確信度を数値として添付する ATMS を提案している．相違点は，添付した数値の意味と計算方法が異なる点と，マルチエージェント環境や分散環境における整合性の維持については考慮されていない点である．文献 [4] では，添付した数値は仮説の確信度，つまり確からしさを表し，計算方法として MYCIN の確信度に基づく方法 [16] を用いている．本システムでは仮説に重要度を添付した．スケジューリングなどの領域で重要度はユーザの個人的な好みを表す．そこで，あるコンテキストに対する重要度の計算方法として，そのコンテキストに含まれる仮説の重要度のうち最大となる重要度を取る方式を提案した．

マルチエージェント環境における整合性維持システムに関して，扱うことが可能なコンテキストの数およびコンテキストの管理手法に基づいた分類を図 6に示す．まず，JTMS をマルチエージェント環境で扱えるように拡張した DTMS[2] がある．DTMS では単一のコンテキストしか扱うことができない．本論文で示したようなスケジューリングなどの領域では，そのコンテキストがユーザにとって最も好ましいかどうか分からず，適当ではない．ATMS をマルチエージェント環境で扱った DATMS[15] も提案されている．DATMS[15] は，全体としては複数のコンテキストを維持できないことが，文献 [13] において示されている．マルチエージェント環境において複数のコンテキストを維持できる整合性維持システムとしては，MXFRMS[13] がある．MXFRMS[13] では，整合性を維持すべきコンテキストにだけフォーカスを当てるという手法でコンテキスト管理を行う．本システムではあるコンテキストにフォーカスを当てるのではなく，各コンテキストに対して好みを表す重要度を計算しコンテキスト間の順序付けを行うことによってコンテキストを管理する．

次に，マルチエージェント環境における整合性維持システムを構築する上で考察すべきであるエージェント間のデータの一貫性についての考察を述べる．一般にエージェント間のデータの一貫性のレベルには，local な一貫性，local&share な一貫性，および global な一貫性が考えられる [2]．local な一貫性は，どのエージェントも内部的には無矛盾であることを意味する．local&share な一貫性では，どのエージェントも内部的に無矛盾であり，エージェントのあるグループ内で共有するすべてのデータには一貫性があることを意味する．global コンシステンシーは，エージェントは内部的にも他のすべてのエージェントとの間にも一貫性があることを意味する．

一般に global な一貫性を維持することはマルチエージェント環境では非現実的と言われており，local&share な一貫性が維持される．既存の DTMS，DATMS，および MXFRMS では，local&share な一貫性が維持されている．本論文で示したように現実的なグループのスケジュールには，local スケジュール，local&share スケジュール，および Global スケジュールが考えられ，global な一貫性を維持することも意味があると考えられる．そこで本システムでは，既存の DTMS，DATMS，および MXFRMS では対象としていない global な一貫性を global なスケジュールに関して管理する．

次に，スケジュールの情報のプライバシーの保護について考察する．本システムでは，エージェントが渡されたデータに対して，信頼度を基に重要度を計算する．そして他のエージェントから渡されたデータをそのまま受け入れることはなく，スケジュールを変更する場合は重要度をチェックしている．あるスケジュールを受け入れるかどうかを決定するのはあくまでも受信したエージェントであり，あるエージェントの持つ情報が他のエージェントから勝手に操作されることはない．つまりユーザのスケジュールの情報に関するプライバシーは保護されていると言える．

マルチエージェントによるスケジューリングに関する関連研究との相違を以下に述べる．文献 [6] では会議のスケジューリングのためのエージェント間の交渉をモ

デル化している．エージェントは constraint-relaxation というプロセスによって，スケジュールに対する好みを表す制約を緩和していく．文献 [7] では，ユーザの好みを用いた自動会議スケジューラーを試作している．ユーザの好みは重要度で表され，ユーザはどこまで妥協できるかを閾値を入力することによって表現する．文献 [6] や文献 [7] と本論文との相違点は，文献 [6] や文献 [7] ではある会議をスケジューリングする際のエージェント間の交渉方式に焦点を当てているが，本論文ではグループのスケジュールの整合性を管理する手法に焦点を当てている点である．

6 おわりに

本論文では，好みに基づく分散 ATMS を用いたグループスケジュール管理システムを提案した．現実的なグループのスケジュールを基にして，スケジュールが影響する範囲に基づいてグループのスケジュールを３つのタイプに分類した．各タイプごとにスケジュールの決定プロセスが異なることに注目し，各タイプごとの整合性管理手法を提案した．特に Local&Share スケジュールでは，重要度に基づく整合性管理手法を示した．分散 ATMS に重要度を導入することによって整合性を維持しながら仮説に順序付けが可能となる．さらに他のユーザの提示する重要度を評価する指標として信頼度を導入した．信頼度によって，極端に大きい重要度を連続して添付するユーザの添付する重要度を低く評価することが可能となり，ユーザの恣意的な重要度の添付を防ぐことができる．本システムの利点は，(1) 分散 ATMS に基づいて，個々のメンバーのスケジュール間の整合性を管理できる点．(2) エージェントが信頼度を基に重要度を計算することによって，メンバーの個人的なスケジュールのプライバシーを保護できる点．(3) スケジュールに重要度を添付することにより個々のメンバーのスケジュールに対する好みの反映が可能な点である．

ユーザの好みを反映するためには，スケジュール全体の変化の度合も重要である．Local&Share または，Global なスケジュールがスケジュールされる度に，個々のユーザのスケジュールが大きく変更されることは望ましくない．そこで，ユーザのスケジュールに対する好みを考慮しながら，なるべく変化の少ないスケジュールを採用する必要がある．好みや意見を変更する際，変更の度合をなるべく少なくする手法については Belief Revision[5] の分野で盛んに行われている．Belief Revision の手法をエージェント間の交渉に導入することによって，ユーザの好みや意見の変更を少なくすることは今後の課題である．

参 考 文 献

[1] Arnold, K. and Gosling, J.: The Java Programming language. Addison-Wesley, 1996.

[2] Bridgeland, D. and Huhns, M., “Distributed truth maintenance,” In Proc. of the 7th National Conf. on Artificial Intelligence (AAAI-90), pp. 72–77, AAAI, 1990.

[3] John de Kleer, “An Assumption-based TMS,” Artificial Intelligence, Vol.28, No.1, pp.127–161, 1986.

[4] 董方清，中川裕志，“不確実な知識における ATMS,” 人工知能学会誌，Vol.3，No.1，1988.

[5] Gardenfors, P. : Belief revision. Cambridge University Press, 1992.

[6] Garrido, L., and Sycara, K.: Multi-agent meeting scheduling: Preliminary experiment results. In Proc. of Second International Conf. on Multi-Agent Systems(ICMAS-96), pp.95–102., 1996.

[7] Haynes, T., Sen, S., Arora, N., and Nadella, R.:An Automated Meeting Scheduling System that Utilizes User Preferences, In Proc. of the First International Conf. on Autonomous Agents (Agents’97), pp. 308–315, 1997.

[8] 石井裕，“グループウェアのデザイン,” 共立出版，1994.

[9] 伊藤孝行，新谷虎松，“分散 ATMS に基づくスケジュール管理システムの実現”，人工知能学会全国大会（第８回）論文集，pp. 273–276,1994.

[10] Takayuki Ito and Toramatsu Shintani, “Persuasion among Agents: An Approach to Implementing a Group Decision Support System Based on Multi-Agent Negotiation,” In Proc. of the Fifteenth International Joint Conf. on Artificial Intelligence (IJCAI-97), Morgan Kaufmann, pp. 592–597, 1997.

[11] 伊藤孝行, 新谷虎松, “グループ代替案選択支援システムにおけるエージェント間の説得機構について,” 電子情報通信学会論文誌 D-II,Vol. J80-D-II, No. 10, pp. 2780–2789, 1997.

[12] 金淵培, “エージェント技術の現状と実用化,” 人工知能学会誌, Vol. 12, No. 6, pp. 850–860 ,1997.

[13] Gerhard K. Kraetzschmar, “Distributed Reason Maintenance for Multiagent Systems,” Lecture Notes in Artificial Intelligence 1229, Springer-Verlag, 1997.

[14] Lange, D. and Chang, D., :IBM Aglets Workbench, Programming Mobile Agents in Java, http://www.trl.ibm.co.jp /aglets /whitepaper.htm, 1996.

[15] Mason, C. L. and Johnson, R.R., “DATMS:A Framework for Distributed Assumption Based Reasoning,” Distributed Artificial Intelligence Vol. II, editted by M. N. Hurns and L. Gasser., pp.293–317, Pitman Publishers London, 1989.

[16] Shortliffe, E.H., （神沼二真，倉科周介訳）,” 医療コンピュータシステム,” 文光堂，1981.

[17] Shintani, T. and Ito. T, “An Architecture for Multi-Agent Negotiation Using Private Preferences in a Meeting Scheduler,” In Proc. of the fifth Pacific Rim International Conf. on Artificial Intelligence (PRICAI-98), 1998 (to appear).

[18] 宇井徹雄, “意思決定支援とグループウェア,” 共立出版, 1995.