なるべく多くの会議開催のための会議日程調整法の考察

学生会員 村上 隆生　　学生会員 平井 和則　　正会員 程 子学
会津大学コンピュータ理工学部

会議日程調整法とは、いままで個々で行なっていた、どの会議に参加するか、またその会議に参加するための時間調整などの問題を分散ネットワーク上でコンピュータを導入することにより、会議に参加する人間の負担を少なくしようという研究である。会議調整の目標は、なるべく多くの会議を開催させる、個人の意思をなるべく多く取り入れることである。本論文では時間-会議間の拡張増加道の考え方を導入することで前者の目標を、時間指向優先順位、会議指向優先順位という2つの要素を導入することにより後者の目標を可能にすることを試みる。

1　はじめに

一般社会では様々な会議が行なわれている。会議とは、複数の人間が参加することで、一人では解決できない様々な問題を解決しようとする試みである。会議に参加する人間は、会議の内容（問題点）を考えることにより多くの時間を使うべきであるが、実際は、どの会議に参加するか、またその会議に参加するための時間調整などの問題にも多くの時間を使っているのが現状である。この論文では、これらの問題をネットワーク上で、分散的に、コンピューターを用いることにより、会議に参加する人間の時間を会議調整の雑務などにとらわれないようにし、それらの人々により多くの時間をあたえ、会議の内容について考える時間を増やそうということを目的としている。

従来の研究 [1][2][3] では、会議をどこの時間に設定するか、その時間をどのように決定するか（ユーザーの意思をどれだけ反映できるか）、が問題となっていたが、なるべく多くの会議を開催させることを考えていなかった。この論文では会議をなるべく多く開催させる、個人の意志をなるべく多く取り入れる、という2つを目標としている。この論文では前者は、「時間-会議間の拡張増加道」を導入することによって、後者は時間指向優先順位（Time-Priority）と会議指向優先順位（会議 Priority）という2つの要素を導入することによって、なるべく多くの会議をできるだけ個人の意思を尊重し開催可能にする会議日程調整法を提案する。

時間-会議間の拡張増加道 とは、一般的にいうと時間の譲り合いのことである。会議時間を決定する時、複数の実行可能時間を持つグループは、一つの実行可能時間しかないグループと会議時間が重複しているときに会議時間を譲ることが可能である。

Priority（優先順位）とは、人が物事を判断する時の一つの基準である。この論文では、時間に対する Priority（Time-Priority）、会議に対する Priority（会議 Priority）

Design of Methods for Scheduling as Many Meetings as Possible, Ryuusei Murakami, Kazunori Hirai, Zixue CHENG

という２つのPriorityをうまく組み合わせることによって、より個人の意思を尊重しようと試みる。

以下、2. で会議日程調整問題を議論する上での、モデルと問題の定義を与え、3. で会議日程調整アルゴリズムを提案する。

2 モデルと問題の定義

2.1 会議日程調整モデル

以下のモデル上で会議日程調整問題を議論する。

- 会議は $C = \{c_1, c_2, ..., c_n\}$ で表し、参加者（会議に参加する可能性のある人）は $P = \{p_1, p_2, ..., p_n\}$ で表す。
- 会議 c_i に参加する可能性のある複数の参加者の集まりを $g_i \subseteq P$ とおく。これは、より一般的に $G = \{g_1, g_2, ..., g_n\}$ と書くことが出来る。
- 会議側から与えられる情報は、その会議に参加する可能性のある参加者 (g_i)、q_i（会議 c_i が会議開催可能になる人数）。

2.2 問題の定義

以下の条件を保持し、各参加者はどの会議に参加するかを決定する問題を解くことを議論する。

1. 一人の参加者が同時に２つ以上の会議に参加することは出来ない。また、２つ以上の会議が、必ず特定の一人の参加者を必要とする時は同時に開催できない。（相互排他性）
2. 参加者の時間 Priority、tp を最優先させ、会議 Priority、mp をできるだけ満足させる。
3. 最低一つは会議が開催される（Deadlock-free）
4. 各会議は一定人数（q_i で表す）以上の会議参加者を必要とする。（最低人数出席条件）
5. できるだけ多くの会議を開催させる。

会議決定アルゴリズムにおいて、個々の参加者は必ず一つの会議にしか参加できない、しかも、その決定は個々の参加者の tp を最優先させ、mp をできるだけ満足させるような結果なので、条件１、２は保証される。また、最低、一つでも会議を開催できるように、条件３を保証する。会議開催のため、最低限の人数が必要になる（一人では開催できない）ため、条件４を保証する。増加道の考え方を用いることにより、より多くの会議が開催可能になる（会議が減ることはない）ため、条件５を保証する。

3 アルゴリズム

この論文では、拡張した増加道の考え方に基づいて、複数の会議の日程調整問題の同期性、相互排他性、Deadlock-free、開催会議数の最大化などの性質を満足させ、また、エージェントを平等に扱うことにより、分散的にシステムを利用できるようにする。その問題を解くための解き方を説明する。

3.1 エージェントの説明

まず、アルゴリズムを考える前にエージェントという概念を考えてみたいと思います。一般社会におけるエージェントとは、個人または団体における代理人として存在し、代理人の仕事とはスケジュール管理または諸処の手続きの代行などと考えられます。この論文の場合、エージェントは各参加者

の会議スケジュール管理を行なう。会議スケジュール管理とは各参加者がどの会議に参加すれば最もよいのかを各参加者にかわり、各エージェントが行なうということです。またエージェントには次の条件がある。

- 各参加者ごとに1つのエージェントを設ける。
- 各エージェントは平等で、特定のリーダーは存在しない。
- 各エージェントの初期設定として次のものがある。
  1. 参加者 p_jの TT_{p_j}及び $mp_{c_i}(p_j)$
  2. 各会議側から与えられる情報。(会議に参加する可能性のある参加者 (g_i)、会議開催可能人数 (q_i)

3.2 基本アイディア

- Time - Table —— 参加者の時間はTime-Tableを用いて考える。1日の時間を時間単位ごとに区切ることで、時間を1つ1つのスロットのようにみることが出来る。Time-Tableとは、その区切られた時間と、その時間へのPriorityとをペアにしたものである。時間単位をt_a(min), Time-Tableのスロットの数をt_nとすると、$t_n = 1440(min)/t_a$で表される。よって参加者 p_jの Time-Table は

$$TT_{p_j} = \{(t_i, tp_{t_i}(p_j))|t_i \in \{t_1, ..., t_n\} , tp_{t_i}(p_j) \in \{1,2,3\}\}$$

ただし t_i とは Time-Table の1つのスロットで

$$t_1 = (t_a * 1)/60$$

$$t_2 = (t_a * 2)/60$$

$$\vdots$$

$$t_i = (t_a * t_i)/60(min)$$

と表される。例えば、t_aを30分とすると、
$t_n = 1440(min)/30(min) = 48$ となり、
$t_1 = (30 * 1)/60 = 0.5$ と書ける。これは
"0：00～0：30" を表すとする。
また、$tp_{t_i}(p_j)$ とは参加者 p_jの時間 t_i に対する Time-Priority を表す。

- Group Time - Table —— まず、c_iの会議時間を決定することを考える。グループ g_iは、まずグループ全体の空き時間を探す必要がある。空き時間が複数個存在する時のことを考え、グループの Time-Table (GTT_{g_i}) を作成する。グループ Time-Table は個人の Time-Table のように全ての時間に Priority をつけるのではなく、全体の空き時間ごとに Priority をつける（個人の Time-Priority の合計が高い順につける、同じ場合は時間の早い方が高くなる)。これは、以下のように簡単にかける。
  - 参加者 p_jの会議参加可能時間を取り出す。

$$TT_{p_j}(l) = \{(t_k, tp_{t_k}(p_j))|(t_k, tp_{t_k}(p_j)) \in TT_{p_j}, tp_{t_k}(p_j) = l\}$$
ただし、$l = 2,3$

– グループ g_iのグループ Time-Table の作成

$$GTT_{g_i} = \{(t_k, \sum_{p_j \in g_i} tp_{t_k}(p_j)) | (t_k, tp_{t_k}(p_j)) \in \cap_{p_j \in g_i} (TT_{p_j}(2) \cup TT_{p_j}(3))\}$$

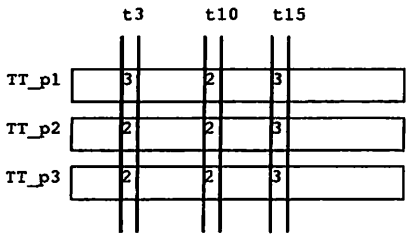

図 1: グループ Time-Table の例

グループ g_iの会議開催可能時間を取りだし、その時間 t_kにおける各参加者 p_jの $tp_{t_k}(p_j)$ の合計を出すことでグループ Time-Table の Priority とする。$tp_{t_k}(p_j)$ の合計の高いものから、対応する t_i をグループ Time-Table の先頭にのせる。(図 1 の場合、$GTT_{g_1} = \{t_{15}, t_3, t_{10}\}$ のように書ける。このとき $\sum_{p_j \in g_i} tp_{t_k}(p_j)$ は $t_{15} > t_3 > t_{10}$ となる)

- 会議指向優先順位 —— 各参加者 p_jによる各会議 c_iへの重要度を会議 Priority、$mp_{c_i}(p_j)$ として表す。
- 時間-会議間の拡張増加道 —— 増加道とは、一般社会でいえば時間の譲り合いともいえる。共通な参加者をもつ 2 つの会議があり同時に開催できない場合、1 つのグループが会議実行時間を複数個持っていて、会議実行可能時間がそのグループと重複している 1 つしかないグループは会議を開催させるために、会議開催可能時間を複数個持っているグループにメッセージを送り、会議時間を譲ってもらう。

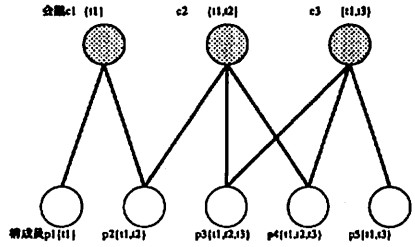

図 2: 時間-会議間の拡張増加道の例

例えば、上の図 2 のような状態の場合を考える。会議 $c1, c2, c3$ が存在する。参加者 $p1, p2, p3, p4, p5$ がそれぞれの会議に参加する可能性がある。各参加者それぞれ $\{t1\}, \{t1, t2\}, \{t1, t2, t3\}, \{t1, t2, t3\}, \{t1, t3\}$ の会議実行可能な空き時間が存在する。これを元にそれぞれのグループ Time-Table を作成すると、$GTT_{g_1} = \{t_1\}, GTT_{g_2} = \{t_1, t_2\}, GTT_{g_3} = \{t_1, t_3\}$ となる (それぞれのグループの tp の合計が $t1$ で一番高いとする)。この場合、全ての会議が t_1において会議時間が決定されることになるので、g_1のメンバーは会議 c_1を通じて会議 c_2の参加者 (g_2) に時間を譲ってもらえないかとメッセージを送る。g_2の参加者が了解したならば、結果は以下の図 3 のようになる (c_3は g_1 の参加者と重なっていないので同じ時間 t_1に会議を開催することができる)。これらの変更をすることにより、より多くの会議が実行できるようになる。

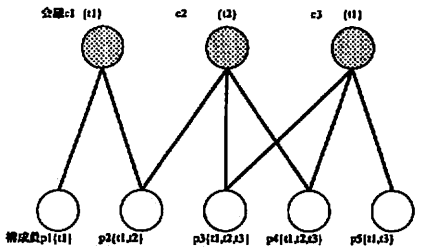

図 3: 時間-会議間の拡張増加道を実行した結果

3.3 会議日程調整アルゴリズム

mp、tp は基本的に 3 段階とし、それぞれ次のような意味合いをもつ。

mp	意味合い	tp	意味合い
1	参加する必要はあまりない	1	不参加
2	参加したほうが良い	2	参加可能
3	参加しなければならない	3	参加

変数名	機能
q_i	会議 c_i における会議開催に必要な最低人数
GTT_{g_i}	グループ g_i の Time-Table
$sum\_mp_{c_i}$	会議 c_i における グループの mp の合計
$s\_time_{c_i}$	会議 c_i における 仮決定会議時間
$m\_time_{c_i}$	会議 c_i における 本決定会議時間

メッセージ名	機能
$maybe\_join_{p_j}$	会議側へ参加者 p_j が参加するかも知れないという意思表示
ask	自分の情報を持っているかの確認
ask_yes ask_no	ask に対する返事。 持っている 持っていない
$give\_time_{t_i}$	t_i の時間を譲って欲しいというメッセージ
$time\_yes$ $time\_no$	$give\_time_{t_i}$ に対する返事。 t_i を譲ります。 t_i を譲りません。
$decision_{c_i}$	グループ g_i に会議 c_i に参加するかどうか質問する。
ok_{c_i} not_{c_i}	$decision_{c_i}$ に対する返事。 参加します。 参加しません。

以下に会議日程調整アルゴリズムを示す。また、会議日程調整アルゴリズムを説明するのに、変数名、メッセージ名の説明を上の表に示す。

1. 初期設定
   (a) 各参加者 p_j のデータ入力：参加する可能性のある会議、その会議における mp、自身の Time-Table。
   (b) 各参加者 p_j は、参加する可能性のある会議へ、メッセージ $maybe\_join_{p_j}$ を送る。また、各会議側はこのメッセージを元にグループ g_i を作成する。
   (c) 各会議側からの情報取得 ： 参加する可能性のある会議のグループの確認、q_i
2. データ送信・受信

(a) 自分が参加する可能性のある会議のグループへ ask のメッセージを送る。

(b) メッセージ ask を受けとった参加者はメッセージ ask_yes、ask_no のどちらかの返事を返す。

- ask_yes のメッセージを受けとったら、1.(a) で入力した情報を送らない。
- ask_no のメッセージを受けとったら、1.(a) で入力した自分の情報（TT, mp）を送る。

(c) 自分以外の全ての参加者から情報を得たら、各グループごとに $sum\_mp_{c_i}$ を出す。

(d) 各グループ g_i の会議開催可能時間をだし、TT_{g_i} の作成。会議開催可能時間の中でグループでの tp の合計が一番高い時間を $s\_time_{c_i}$ とする。$s\_time_{c_i}$、GTT_{g_i} を各グループ g_i に送る。

3. 会議仮決定

(a) 自分が参加する可能性のある会議の $s\_time_{c_i}$ における tp が、$tp = 2 \wedge (mp < tp)$ のとき、その会議には参加しない。

(b) 複数の会議の $s\_time_{c_i}$ が同じ時間の時、自分が参加する可能性のある会議の mp の最大値を探す。最大値が 1 つのとき、その会議へ参加決定。

(c) mp の最大値が複数存在するとき、$sum\_mp_{c_i} \times q_i$ の比較により値の大きい方の会議へ参加する。

(d) それでも決まらない時はランダムによる決定。

4. 結果決定

(a) 決定アルゴリズムにより決定した参加する会議をグループ g_i に送る。

(b) ここで、時間-会議間の拡張増加道による、会議数増加を試みることが出来る。ある参加者が時間を譲って欲しい場合、その譲ってもらいたい時間を持つグループにメッセージ $give\_time_{t_i}$ を出す。

- 相手の返事が $time\_yes$ のとき — t_i は譲ってもらえ、会議開催可能となる。
- 相手の返事が $time\_no$ のとき — t_i は譲ってもらえない。

(c) 4.(b) の結果をグループ g_i に送り、各参加者にメッセージ $decision_{c_i}$ を送る。$decision_{c_i}$ を受けとった参加者は ok_{c_i}、または not_{c_i} を送らねばならない。

(d) $ok_{c_i} \geq q_i$ の（会議最低必要人数が集まった）ときは、会議開催決定。$not_{c_i} > q_i$ のときは、グループ g_i での参加者が一番少ない会議を消して、もう一度 3 へ。

4 むすび

本論文では、時間-会議間の拡張増加道、2 つの優先順位（時間指向、会議指向）を導入し、それに基づいた会議日程調整法を提案した。

今後の課題として、以下のようなものがあげられる。

1. 会議日程調整方法の妥当性、メッセージ複雑度などの評価。
2. 会議日程調整システムの作成。

3. ヒューマンインターフェースの強化。

4. ランダムによる会議決定などでは、参加者の意思をあまり反映できなくなるので、よりユーザーの意思を反映できる方法を開発する。

参考文献

[1] Eithan Ephrati, Gilad Zlotkin, Jeffrey S.Rosenschein;"Meet Your Destiny:A Non-Manupulable Meeting Scheduler"

[2] Zixue CHENG;"A Decentralized Social Algorithm for Commitee Coordination Problem"

[3] Kazuo Sugihara, Tohru Kikuno, Noriyoshi Yoshida; "A Meeting Scheduler for Office Automation"