Vol.2011-AL-134 No.8
2011/3/7

カンファレンスプログラム編成のための局所探索法の改良

上原康史[†1]　田岡智志[†1]　渡邉敏正[†1]

カンファレンスとは学術的な研究発表会や国際会議などの総称であり，実施までに多くのプロセスを経る．その際，数十件から数千件の投稿論文を扱うため，セッションの編成にも多くの労力と時間がかかる．セッション編成は，組合せ的な制約条件（同じ時間帯での著者・座長の重複の禁止等）を満たしつつ評価値を最適化する組合せ最適化問題の一種であり， NP-困難であることが知られている．筆者らは，組合せ最適化問題に対する発見的解法として知られている局所探索法 (Local Search) に着目した解法を提案した．本稿では，その解法の改良およびアニーリング法 (Simulated Annealing) を用いた解法を提案し，計算機実験によりそれらの有用性を示す．

Improving Local Search Algorithms for Organizing Conference Programs

Yasufumi Uehara,[†1] Satoshi Taoka[†1]
and Toshimasa Watanabe[†1]

The term “Conference” means an academic meeting in general for presentation and discussion on scientific research/education activities, possibly including refereeing procedures. There are many intermediate processing steps before the completion. Among them session scheduling takes much effort and long processing time, because several dozen to thousand papers may have to be handled. It is a combinatorial optimization problem for maximize or minimize a given objective function under some constraints, such as avoiding assignments of chairpersons to the time slots for presentation of their papers. The problem is known to be NP-hard. The authors have algorithms proposed based on the local search method, well-known as heuristics for combinatorial optimization. In this research, we improve the algorithm, propose a new algorithm by simulated annealing, and evaluate their capability through computing experiment.

1. はじめに

カンファレンス（学会）とは，様々な立場の人々が研究の経過や成果を発表し，今後の課題や問題点などを議論する場である．開催には論文投稿から査読，採否決定と通知，セッションスケジューリングなどをはじめ，非常に多くの時間と労力を要する作業が必要となる．その業務の中で，セッションスケジューリングはカンファレンスを運営するうえで最も困難な作業のひとつであり，運営側は多大な労力と時間をかけて行わなければならない．発表件数が多いほど，より多くの会場を利用した並列的発表進行が必要となり，編成作業は一層困難な作業となる．

セッションスケジューリング問題は組合せ的な制約条件を満たしつつ評価値を最適化する組合せ最適化問題[1)] の一種である．一般的に，カンファレンスにおける制約条件には論文の著者（複数名の場合が多い）や，論文の発表内容などに関わるものが多い．さらに発表者だけでなく，座長や聴講者に関する制約条件も複数存在する．また論文投稿者による入力に矛盾や誤りが存在することも多く，それらを発見するという前処理だけでも時間のかかる作業であり，その負担の軽減も必要である．

当研究室では，カンファレンス運営の総合的支援システム AllConf[2)] を実用化しており，現在まで既に多くのカンファレンスにおいて使用された実績がある．

関連する研究として 3)–5) がある．3) はグルーピング GA を利用した方法で，論文のキーワードに基づいた分類からセッションスケジューリングを行なっているが，制約条件の考慮が不十分である．4) は貪欲法と反復改善法を利用した方法で，論文のキーワードや著者の類似性 6)，7) を定義し，セッションスケジューリングを行っている．5) は上記の類似性を使用して，整数計画法と動的計画法を利用した方法で，セッション（分野）の配置までは行っているが，セッション内の発表順の決定には言及されていない．また，求解に非常に時間がかかっている．

現実的要求の観点から見てこれらの既存解法で不十分な点は，連続講演希望への対応などの制約条件の充実である．迅速に反復改善作業を行うなどのために計算の高速化も重要である．筆者らは，組合せ最適化問題に対する発見的解法として知られている局所探索法 (Local Search) に着目して，実用上十分な制約条件を設定してより多くの場面で使用できる

†1 広島大学大学院工学研究科
Graduate School of Engineering, Hiroshima University

柔軟性を有する解法を提案した。[8] 本稿では，その解法の改良，アニーリング法 (Simulated Annealing) を用いた解法を提案し，計算機実験によりそれらの有用性を示す．

2. セッションスケジューリング

セッションスケジューリングとは，論文の講演者や著者，カンファレンスの実施会場などに関するデータを使用して，ハード/ソフト制約 (2.2 節参照) などを考慮しながらカンファレンスプログラムを作成する作業である．

時間割編成の最も一般的な例として学校における授業時間割編成が挙げられる． しかし，授業時間割編成には存在しないが，カンファレンスにおけるセッションスケジューリングでは重要である条件が存在するなど，異なる点も多い．

共著者という概念，空き時間の有無などに関わる制約条件の存在などがその主なものである．それら全てを考慮してセッションスケジューリングを行う能力を有し，かつ様々なカンファレンススケジューリングに対応したシステムの必要性は高い．

2.1 対象とするカンファレンスの特徴

対象カンファレンスは以下の運営状況 (1)〜(4) を想定する．

(1) 研究会や分科会ごとではなく，幹事やその他役員が一括して全体のセッションスケジューリングを行う
(2) ある論文発表の直前または直後に発表したい，という著者からの希望（連続講演希望と呼ぶ）へ対応する
(3) 開催側が複数の分野名をあらかじめ提示し，各投稿者が自身の論文の分野を選択する
(4) 発表する論文をどの分野に属するものとするか，について第 1〜第 x 希望までの指定へ対応する

このような特徴は主に学会支部などの連合大会などに見られ，必ずしも全カンファレンスに共通するものではない．運営状況 1 に見られるように，一括して全体のセッションスケジューリングを行うことは労力と時間を費やす作業である．加えて実施日数が少ない場合や発表件数が数百〜数千件となる場合も存在するため，それらに対してセッションスケジューリングを行うと，解決すべき様々な課題に直面することが多い．

2.2 制約条件[8]

カンファレンスでの一般的な制約条件として，以下のようなものが挙げられる．ここで，ハード制約とは必ず満たさなければならない条件 (必須条件) であり，ソフト制約とは満たすならば不便・不都合を解消できる条件である．

2.2.1 ハード制約

H1 会場数は，上限を超えてはならない
H2 全論文を日程に組み込む
H3 各連続講演希望グループは指定された順番に連続して配置する
（ただし，複数セッションに跨って配置しても良い）
H4 講演者が同じ論文は同一時間帯に重複して配置しない
H5 各セッションは希望講演分野の 1 つが同じ論文で構成する
H6 各セッションの論文配置数は（$l+1$）件を上限とする
H7 セッション間の休憩時間は 9 分〜18 分，昼休憩は 50 分以上確保し，昼休憩後の開始時間は統一する
H8 座長は同じ時間帯に重複して配置しない
H9 座長と同一所属の論文はそのセッションに配置しない
H10 座長が著者である論文はそのセッションに配置しない
H11 （H9 以外にも）座長が著者である論文がある場合は，それらは座長を務める時間帯の別会場に配置しない

なお，座長の割当は全論文をセッション表に配置した後に行うものとし，本稿では H1〜H7 のみをハード制約とする．

2.2.2 ソフト制約

論文に関する制約条件

SR1 同じ分野の論文は同じ会場に連続して配置する
SR2 （事前に指定される）似た内容の分野同士を別のセッションとして同じ時間帯に配置しない
SR3 同一セッションに配置された論文に対し，同一著者の論文は連続して配置する
SR4 著者が同じ複数の論文を同じ時間帯の別セッションに配置することを避ける
SR5 一つのセッションに同一著者の論文をできるだけ少なく配置する
SR6 遠方から来る発表者の論文は，（発表者が）除外指定した時間枠があればそこへの配置を避ける
SR7 連続講演希望により指定された論文同士はセッションを跨いで配置しない

セッションに関する制約条件

SS1 空のセッションは，開催初日の最初のセッションに配置する
SS2 セッションの開始時間をできるだけ揃える

SS3　各セッションに論文は l 件を超えて配置しない

上記のいずれかの制約に違反している論文/セッションを，制約違反論文/セッションと呼ぶ．ソフト制約条件は互いに競合しやすいため各制約条件に対して，優先度と違反の発生しやすさに応じた違反値を定めている．この違反値の合計を解の評価方法とする．なお，ハード制約には違反値を付加せず，求解手順においてどのソフト制約よりも優先する．

2.3 編成の準備

セッションスケジューリングの作業に入る前の準備として,「データの整合性を確認して適切なデータ集合に修正する」といった作業が必要になる場合がある．投稿者自身が様々な情報を登録していく際に，誤った情報や矛盾した内容を入力してしまう可能性がある．そのため，セッションスケジューリングを行う前にこれらの入力ミスやデータに内在する矛盾を発見し修正する必要がある．

2.4 編成の方法

セッションスケジューリングの作業は，最初にいくつかの制約条件を考慮しながら初期解を生成し，その後に他の制約条件を満たすよう論文の再配置を繰り返す．投稿論文数が数百件を数えるカンファレンスでは，参照すべきデータの数は数千に及ぶ．セッション表と各種データを交互に参照し，それらを逐一確かめながらセッションスケジューリングを行うことは長時間を費やす作業である．これら一連の作業を，高速で高精度の発見的解法を利用して自動化し，作業の負担を軽減することを目指す

3. 諸　定　義

開催日数 $= d$，開催 i 日目に使用する**会場数** $= h_i$ $(i = 1, \ldots, d)$，開催 i 日目に使用する**セッション** (時間枠) **数** $= s_i$ $(i = 1, \ldots, d)$, **セッション内の基本論文数** $= l$ (ただし，$l + 1$ まで許す場合がある) であるカンファレンスを対象とする．

論文情報とは著者によって入力された発表者名，著者名，連絡先，希望講演分野（複数入力可），連続講演希望などを指す．

$\mathbf{h} = \{h_1, \ldots, h_d\}$，$\mathbf{s} = \{s_1, \ldots, s_d\}$ とする．論文を割当てるマス目の集合を**セッション表** M と呼ぶ．セッション表 M における，a 日目 第 b 会場 第 c セッションの k 番目のマス目を $M(a, b, c, k)$ と表す．また，a 日目，第 b 会場，第 c セッションのマス目集合を $M(a, b, c)$ と表す．また，$M(a, b, c, k)$ に論文が配置されていない場合，$M(a, b, c, k) = -1$ とおく．M のマス目の集合に，a, b, c, k をそれぞれ長さ 4 の語'abck' と考えた辞書式順序集合 $\leq_r$ を導入して全順序集合 $(M, \leq_r)$ を考える．全ての論文が割り当てられ，かつハード制約を全て満たすセッション表を**実行可能解**と呼び，M_f と表す．論文やセッションに関する制約条件に違反値を設定し，実行可能解 M_P の評価値を $Eval(M_P)$，違反数を $Num(M_P)$ と表す．その他の定義は紙面の都合上省略する．

4. 局所探索法の概要

組み合わせ最適化問題に対し，いくつかの発見的解法が知られている．例えば，アニーリング法 (Simulated Annealing, SA)，タブー探索法 (Tabu Search, TS)，遺伝的アルゴリズム (Genetic Algorithm, GA) などがある．本研究では，組合せ最適化問題に対する最も基本的な枠組みである局所探索法 (Local Search, LS) に着目する．9) では LS をメタ戦略の一般化として捉えることで，これらの発見的解法を系統的にまとめている．そこで，本稿では 9) の局所探索法の定義を用いることにする．局所探索法の概要を述べる．

局所探索法：　与えられた初期解 M から始め，近傍 $\aleph_X(M)$ 内の解に一定のルールで移動する操作を，局所探索の終了条件がみたされるまで反復する．

局所探索法の動作を定めるには以下を決める必要がある．

- 初期解
- 近傍の定義
- 解の評価方法
- 移動戦略 (解の更新方法)
- 終了条件

5. 既存解法[8)]

本章では，8) で提案されている解法の概略を示す．

5.1 初期解の生成

ハード制約のうち H2，H3，H5，H6, H7 を満たし，優先度が高く比較的調整し難いソフト制約 SR1 を出来る限り満たす解を求め，それを初期解とする．ただし，空のセッションはなるべく開催初日の早い時間に配置する．

5.2 近傍の定義

近傍の定義は局所探索法において極めて重要である．近傍は様々な定義が考えられる．例えば，交換近傍，挿入近傍などがある．8) では交換近傍を定義する 3 種類のルール A, B, C を定義している (図 1)．そしていずれのルールにおいても初期解制約 (5.1 節参照) を満たした状態を維持し，さらに連続講演希望順を崩さないようにしている．

ルール A　解 M において $M(a,b,c,k)$ を一つ選択し，$M(a,b,c,k)$ と異なる論文 $M(d,e,f,j)$ と入れ替える．

ルール B　解 M において，連続講演希望グループ $\{M(a,b,c,k),\ldots,M(a,b,c',k')\}$ を選択し，これと要素数が等しく，かつ $\{M(a,b,c,k),\ldots,M(a,b,c',k')\}\cap\{M(d,e,f,j),\ldots,M(d,e,f',j')\}=\emptyset$ である連続して配置されている論文集合 $\{M(d,e,f,j),\ldots,M(d,e,f',j')\}$ と論文の並びを保持して入れ替える．

ルール C　解 M において，セッションの論文集合 $M(a,b,c)$ を選び，$M(a,b,c)$ と異なるセッションの論文集合 $M(d,e,f)$ と，論文の並びを保持して入れ替える．

5.3 解の評価方法

ソフト制約条件は互いに競合しやすいため各制約条件に対して，優先度と違反の発生しやすさに応じた違反値を定める．

5.4 移 動 戦 略

近傍 $\aleph_X(M)$ 内には，一般に改善解が複数個存在するので，**移動戦略** (move strategy) により解の選択方法を定める．代表的な移動戦略として, 次の 2 つがある．

即時移動戦略 (first admissible move strategy)　近傍内で最初に発見した改善解に移動する．

最良移動戦略 (best admissible move strategy)　近傍内を全て探索し, 最良の改善解に移動する．

8) ではこれらの移動戦略に基づき 2 つの解法が提案され，即時移動戦略を用いた解法を LSF (Local Search based on First admissible move strategy)，最良移動戦略を用いた解法を LSB (Local Search based on Best admissible move strategy) とと表記する (局所探索法の定義を変更したことにより，8) のアルゴリズム表記を変更していることに注意されたい)．

5.5 終 了 条 件

近傍内に改善解が存在しなくなったら終了する．

5.6 既存解法 LSF

既存解法 LSF の概要を以下に示す．ここで，$\alpha\in\{A,B,C\}$ とする．紙面の都合上，LSB の詳細は省略する．

$LSF(\alpha)$

入力：論文情報，d，$\mathbf{h}$，$\mathbf{s}$，l

出力：$M_f(X)$ ($\mathcal{N}$ ール X 使用後の M_f)，$Eval(M_f(X))$

1　初期解 M_0 を生成し，$M\leftarrow M_0$ とする;

2　順序 $\leq_r$ に従って小さい順に，各 $M(a,b,c,k)$ に対して Step 2.1〜2.3 を繰り返す．

2.1　$K_{old}\leftarrow\emptyset$ とする;

2.2　次の操作で交換元の論文集合 K_{old} を定める;

$\alpha=A$ **の場合**

$M(a,b,c,k)$ $(\neq -1)$ が制約違反論文であり，かつ連続講演希望グループに含まれていないならば，$K_{old}\leftarrow\{M(a,b,c,k)\}$;

$\alpha=B$ **の場合**

$M(a,b,c,k)$ $(\neq -1)$ が連続講演希望グループ $\mathbf{P}_i$ の先頭要素として含まれており，かつ $\mathbf{P}_i$ 内に制約違反論文が含まれるならば，$K_{old}\leftarrow\{M(a,b,c,k),\ldots,M(a,b,c,k+|\mathbf{P}_i|-1)\}$;

$\alpha=C$ **の場合**

$M(a,b,c,k)$ がセッション $M(a,b,c)$ の先頭要素 (つまり $k=1$) として含まれており，かつ $M(a,b,c)$ が制約違反を起こしているか，制約違反論文を含んでいるならば，$K_{old}\leftarrow M(a,b,c)$;

2.3　もし $K_{old}\neq\emptyset$ ならば，順序 $\leq_r$ に従って小さい順に，各 $M(a',b',c',k')$ に対して Step 2.3.1〜2.3.3 を繰り返す．

2.3.1　$K_{new}\leftarrow\emptyset$ とする;

2.3.2　次の操作で交換先の論文集合 K_{new} $(|K_{old}|=|K_{new}|)$ を定める;

$\alpha=A$ **の場合**

$M(a',b',c',k')$ $(\neq -1)$ が連続講演希望グループに含まれておらず，かつ $\{M(a',b',c',k')\}\neq K_{old}$ ならば，$K_{new}\leftarrow\{M(a',b',c',k')\}$;

$\alpha=B$ **の場合**

以下の条件を全て満たす $\mathbf{P}'$ が存在するならば，$K_{new}\leftarrow\mathbf{P}'$;

- $|\mathbf{P}'|=|\mathbf{P}_i|$ なる論文集合 $\mathbf{P}'=\{M(a',b',c',\ k'),\ldots,M(a',b',c',k'+|K_{old}|-1)\}$;
  ただし，$\mathbf{P}'$ の各要素は -1 ではない．
- $\mathbf{P}'\cap K_{old}=\emptyset$;
- ある連続講演希望グループ $\mathbf{P}_j$ が $\mathbf{P}'\cap\mathbf{P}_j\neq\emptyset$ ならば $\mathbf{P}_j\subseteq\mathbf{P}'$ ($\mathbf{P}_j$ が $\mathbf{P}'$ に含まれる);

$\alpha=C$ **の場合**

$M(a',b',c',k')$ がセッション $M(a',b',c')$ の先頭要素 (つまり $k'=1$) として含まれており，かつ $M(a',b',c') \neq K_{old}$ ならば $K_{new} \leftarrow M(a',b',c')$;

2.3.3 もし $K_{new} \neq \emptyset$ ならば，M において論文の並びを保持したまま K_{old} と K_{new} を入れ替えた解を M' とし，$Eval(M') < Eval(M)$ ならば $M \leftarrow M'$ として Step 2 へ進む;

なお，LSF, LSB で利用した近傍 (5.2 節で定義) の違いを明確にするために次のように表記する．ただし，$\{X,Y,Z\}$=$\{A,B,C\}$ とする．

- 近傍探索の際にルール X を適用する解法 LSF_X，LSB_X
- 近傍探索の際に 3 つのルール X,Y,Z をこの順に適用する解法 LSF_{XYZ}，LFB_{XYZ}

5.7 既存解法 LSF[8] の問題点

LSF では，局所最適解に捕捉されることについて考慮されていない．その理由として，例えば，交換近傍のみであること，改善解のみに移動すること，改善解に移動する毎に順序に従い最初から探索を再開するなどが挙げられる．局所探索法において近傍の定義は最も重要なものの一つで，交換近傍以外の近傍も検討すべきである．また，改善解のみに移動する戦略では局所最適解に到達すると探索は終了する．一般的に，局所最適解の周辺にはより良い解が潜んでいる可能性が高いので，改悪解への移動も許す移動戦略が必要である．探索順序に関しては，近傍の探索に偏りが生じてしまうと解の中でも到達しにくいものができてしまう可能性がある．

6. 提 案 解 法

本章では，5.7 節であげた既存解法の主な問題点である，近傍，探索順序，移動戦略について検討し，その問題点を改善した解法を提案する．

6.1 挿入近傍の追加

5.2 節の交換近傍のみでは違反を解消できない場合がある．そこで新たな近傍を導入し，既存の近傍と組み合わせることによってさらなる解精度向上を目指す．そこで，次に挿入近傍を提案する (図 1).

ルール D 解 M において論文 $M(a,b,c,k)$ を一つ選択し，$M(a,b,c,k)$ と同一会場，同一分野の別論文 $M(a,b,c',k')$ の位置に挿入する．$M(a,b,c,k)$ の位置には論文の並びを保持してひとつずつ詰める．

これを 5.6 節で示した $LSF(\alpha)$ の Step 2.2 と 2.3.2 に組み込んだ解法において，利用したルールの違いを明確にするために次のように表記する．ただし，$\{W,X,Y,Z\}$=$\{A,B,C,D\}$ とする．

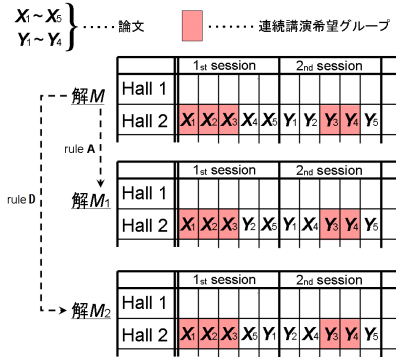

図 1 解 $M_1 \in \aleph_A(M)$，および解 $M_2 \in \aleph_D(M)$ の例.
Fig. 1 An example of solutions $M_1 \in \aleph_A(M)$ and $M_2 \in \aleph_D(M)$

- 近傍探索の際にルール X を適用する解法 LSF_X, LSB_X
- 近傍探索の際に 4 つのルール W,X,Y,Z をこの順に適用する解法 LSF_{WXYZ}, LSB_{WXYZ}

6.2 $LSF+$

LSF は，1 度の交換を行った後に評価値を計算し，改善するかを調べているが，2 度の連鎖的交換により改善できることもある．そこで，以下に示す 2 度の連鎖的交換を行う解法 ($LSF+$ と表記) を提案する．

LSF+(α)
入力：論文情報，d，$\mathbf{h}$，$\mathbf{s}$，l
出力：$M_f(X)$ (近傍ルール X 使用後の M_f)，$Eval(M_f(X))$
1 初期解 M_0 を生成し，$M \leftarrow M_0$ とする．また，$t \leftarrow t_0$ とする．
2 順序 $\leq_r$ に従って小さい順に，各 M に対して a ～ c を繰り返す.
2.1 近傍 $\aleph_X(M)$ 内の解をひとつ選択して M' とおく.
2.2 $\Delta \leftarrow Eval(M') - Eval(M)$ とする.
2.3 $\Delta \leq 0$ ならば $M \leftarrow M'$ として Step 2 に戻る．そうでなければ Step2.3.1～2.3.3 を繰り返す.
2.3.1 近傍 $\aleph_X(M')$ 内の解をひとつ選択して M'' とおく.
2.3.2 $\Delta' \leftarrow Eval(M'') - Eval(M')$ とする.

2.3.3 $\Delta - \Delta' \leq 0$ ならば $M \leftarrow M''$ として Step 2 に戻る．

3 Step 2 で解の更新がなければ暫定解を出力して探索を終了する．そうでなければ，Step 2 に戻る．

ただし，2 度の連鎖的交換を行うと近傍数が莫大に増大するので，以下を考慮して計算時間を抑える．

- 通常の LSF で改善できなくなったときに，2 度の連鎖的交換を行う探索を開始する
- 2 度目の交換を行うのは，1 度目の交換で影響が及ぶ範囲に限定する

6.3 探索順序の改良

局所探索法において，即時移動戦略では近傍内の探索順序が性能に影響を与える． LSF では，順序 $\leq_r$ に従って，近傍解の探索を行い，改善解に移動する毎に順序 $\leq_r$ の一番小さいものから探索を再開していた．そのような探索順序では近傍内の探索に偏りが生じてしまう可能性がある．そこで，探索の偏りを減らすために改善解へ移動後に探索を再開する位置の違いにより次の 3 つの手法 $LSF\text{-}CN$, $LSF\text{-}NS$, $LSF\text{-}NR$ を提案する．

- 改善解に移動後，最初に戻らずにその位置から探索を再開する: $LSF\text{-}CN$(continue)
- 改善解に移動後，交換元の次のセッションから探索を再開する: $LSF\text{-}NS$(next session)
- 改善解に移動後，交換元の次の会場の先頭のセッションから探索を再開する: $LSF\text{-}NR$(next room)

なお，既存解法 LSF では改善解に移動後，最初に戻って探索を再開し，最初から最後まで 1 度も改善しなかったときに終了する．$LSF\text{-}CN$, $LSF\text{-}NS$, $LSF\text{-}NR$ においては，探索が最後まで進むと，もう一度，最初に戻り最後まで 1 度も改善が行われなければ終了とする．具体的には，5.6 節のアルゴリズムの最後に次の記述を追加する．

3 Step 2 において，一度も解の移動が行われなかったとき終了する．そうでなければ，Step 2 へ進む．

6.3.1 $LSF\text{-}CN$

5.6 節の **2.3.3** を次のように変更して，改善解への移動が行われるとその次の交換先候補から探索を再開する．

もし $K_{new} \neq \emptyset$ ならば，M において論文の並びを保持したまま K_{old} と K_{new} を入れ替えた解を M' とし，$Eval(M') < Eval(M)$ ならば $M \leftarrow M'$ として Step 2.3 へ進む．交換元の論文集合の候補は $M(a, b, c+1, 1)$ から，交換先の論文集合の候補は $M(a', b', c', k'+1)$ から順序 $\leq_r$ に従って小さい順とする;

6.3.2 $LSF\text{-}NS$

5.6 節の **2.3.3** を次のように変更して，改善解への移動が行われると交換元の次のセッションから探索を再開する．

もし $K_{new} \neq \emptyset$ ならば，M において論文の並びを保持したまま K_{old} と K_{new} を入れ替えた解を M' とし，$Eval(M') < Eval(M)$ ならば $M \leftarrow M'$ として Step 2 へ進む．交換元の論文集合の候補は $M(a, b, c+1, 1)$ から順序 $\leq_r$ に従って小さい順とする;

6.3.3 $LSF\text{-}NR$

5.6 節の **2.3.3** を次のように変更して，改善解への移動が行われると交換元の次の会場から探索を再開する．

もし $K_{new} \neq \emptyset$ ならば，M において論文の並びを保持したまま K_{old} と K_{new} を入れ替えた解を M' とし， $Eval(M') < Eval(M)$ ならば $M \leftarrow M'$ として Step 2 へ進む．交換元の論文集合の候補は $M(a, b+1, 1, 1)$ から順序 $\leq_r$ に従って小さい順とする;

6.4 アニーリング法

改善解のみに移動する戦略では局所最適解しか得られないという欠点がある．そこで，アニーリング法[10](Simulated Annealing, 以下 SA) を導入する． SA は現在の解 M の近傍 $\aleph_X(M)$ 内の各解 M' に，解の良さに応じた移動確率 (良い解ほど移行しやすい) を設定し，それにしたがって次の解を選ぶ．改悪解であっても移動する確率を与えることにより，局所最適解からの脱出を図る．また，SA は移動確率を温度 t というパラメータによって制御する．以下に SA のアルゴリズムの概要を示す．ここで，$X \in \{A, B, C\}$ とする

Simulated Annealing(α)

入力：論文情報，d，$\mathbf{h}$，$\mathbf{s}$，l, $t_0(>0)$, $t_{\min}(>0)$, $\beta(0<\beta<1)$
出力：$M_f(X)$ (近傍ルール X 使用後の M_f)，$Eval(M_f(X))$

1 初期解 M_0 を生成し，$M \leftarrow M_0$ とする．また，$t \leftarrow t_0$ とする．

2 順序 $\leq_r$ に従って小さい順に探索順序を定め，Step2.1〜2.3 を繰り返す．

2.1 近傍 $\aleph_X(M)$ 内の解を順序に従いひとつ選択して M' とおく．

2.2 $\Delta \leftarrow Eval(M') - Eval(M)$ とする (M' が改悪解ならば $\Delta > 0$)．

2.3 $\Delta \leq 0$ ならば $M \leftarrow M'$，そうでなければ確率 $e^{-\Delta/t}$ で $M \leftarrow M'$ とする．

3 $t < t_{\min}$ ならば暫定解を出力して探索を終了する．そうでなければ，$t \leftarrow \beta \cdot t$ の後 Step2 に戻る．

Vol.2011-AL-134 No.8
2011/3/7

$e^{-\Delta/t}$ は評価値の増加 Δ が大きければ移動確率が指数関数で小さくなることを意味する．$t \to \infty$ のとき評価値にかかわらず移動するランダムウォークとなる．温度が低下すると，評価値を増加させる移動が少なくなり，$t \to 0$ のとき，改善解のみへの移動を許す LS と一致する． SA ではこれを高い温度から開始し，徐々に温度を下げていくことで局所最適解に容易に捕捉されにくい探索を行う．

SA では初期温度，終了温度，温度の更新方法を設定する必要がある．温度の更新方法は冷却スケジュールと呼ばれ，様々な方法が提案されている．本稿では，簡単で，かつ一定の性能が期待できる幾何冷却法 (geometric cooling) と呼ばれる方法を利用する．これはパラメータ $\beta(0 < \beta < 1)$ を用意し，$t := \beta \cdot t$ とする方法である．

7. 実験結果と考察

7.1 実 験 概 要

計算機（CPU: Opteron 285 (2.5GHz), OS: FreeBSD 7.3-RELEASE）上で Perl 5.8.9 により実装した．平成 17 年度から平成 22 年度のある大会の実データに対して解の評価値，計算時間の比較を行った．表 1 と 2 は，既存解法 LSF と LSB と提案解法 LSF-CN, LSF-NS, LSF-NR に関する実験結果を示している．ただし，表 1 は近傍ルールとして既存の A, B, C を利用した場合の結果で，表 2 は提案した近傍ルール D を加えた場合の結果である．

なお，SA と $LSF+$ に関しては十分な計算機実験を行っていないが，一部のデータに対する結果を示す．SA は 9000 秒程度の時間をかけて他の解法よりも 1 %程度良い解を得ることがあるが，多くの場合は時間をかけても改善されなかった．$LSF+$ に関しては，計算時間はかかるが，比較的良い解が得られることがある．LSF と比較して，評価値が約 4 %改善し，計算時間は約 60 倍程度となった．

7.2 考　　察

まず，3 種類の近傍ルールを用いた解法 (表 1) について述べる．LSF において BCA が安定して良い結果が得られていたが，提案解法においても同様な傾向である．これによりは，近傍の少ないルールから多いルールを適用する方が効率よく探索できていると考えられる．計算時間においては，3 つの提案解法は全て既存解法よりも高速であり，なかでも LSF-CN が最も高速である．LSF-CN は既存解法よりも計算時間が平均して 76 %程度短くなった．既存解法では探索が進むにつれて，最初に探索される部分は改善が難しくなるにもかかわらず探索し続けるため，無駄な探索が多い．提案解法ではそのような無駄な探索が減ったことが高速化の要因と考えられる．

表 1 近傍ルール A,B,C を用いたときの平均評価値 eval と平均計算時間 time
Table 1 Average evaluation values: “eval” and average CPU time: “time” when using neighborhood rules A, B and C

	LSF		LSF-CN		LSF-NS		LSF-NR		LSB	
	eval	time	eval	time	eval	time	eval	time	eval	time
ABC	189.0	77.6	188.7	17.7	186.8	19.4	185.2	23.5	185.5	165.6
ACB	188.7	78.5	188.7	18.6	186.3	19.7	184.5	23.6	186.5	168.7
BAC	174.0	63.1	175.0	18.7	175.0	21.3	161.3	22.5	178.8	152.4
BCA	163.8	128.2	168.7	28.5	165.0	31.3	154.0	32.7	164.2	203.0
CAB	171.5	138.9	177.2	33.8	164.5	33.8	162.2	37.5	169.3	234.7
CBA	172.0	137.7	174.5	35.8	161.8	35.2	168.3	38.7	168.5	224.7

4 種類の近傍ルールを適用した解法 (表 2) では $BCAD$ が最も良かった．これも上述した通りの理由であると考えられる．ただし，近傍ルール D を加えたことによる効果はさほど見られなかった．これは，すでに 3 つの近傍ルールを適用している状態では，近傍操作による改善の余地がほとんどなくなっているためと考えられる．

全体として，近傍ルールに関しては $BCAD$，探索順序に関しては LSF-NS，LSF-NR が良いと言える．計算時間では LS-CN に若干劣るものの，LSF-NS，LSF-NR のどちらも LSF より 70 %程度高速化できており，評価値は LSF に比べて 6 %程度良くなった．

8. まとめと今後の課題

本稿ではカンファレンスにおけるセッションスケジューリングに着目し，既存解法の改良と新たにアニーリング法の提案を行った．既存解法の改良では，探索順序を検討することで高速化に成功した．今後の課題として，アニーリング法のパラメータ調整方法の検討，他の発見的解法との比較実験，実験データの充実などがある．

また，現在は 2.1 節で示したカンファレンスに限定しているが，今後様々なカンファレンスで利用できるように次の状況に対応する予定である: 全論文を例えば分野ごと分割して、それぞれで論文の並びを決定し，そして全セッションのスケジューリングを行う，キーワードを元にセッションスケジューリングを行う，など．

参 考 文 献

1) Sait, S.M. and Youssef, H.: *Iterative Computer Algorithms with Applications in Engineering: Solving Combinatorial Optimization Problems*, IEEE Computer Soci-

表 2 近傍ルール A,B,C,D を用いたときの平均評価値 eval と平均計算時間 time
Table 2 Average evaluation values: "eval" and average CPU time: "time" when using neighborhood rules A, B, C and D

	LSF		*LSF-CN*		*LSF-NS*		*LSF-NR*		*LSB*	
	eval	time	eval	time	eval	time	eval	time	eval	time
ABCD	182.3	96.8	187.8	30.7	181.5	37.4	184.3	38.2	184.8	179.2
ABDC	184.7	97.2	188.2	30.7	184.0	35.3	185.0	38.9	184.8	178.7
ACBD	184.2	97.9	187.3	31.3	180.8	37.4	183.2	39.0	186.2	179.1
ACDB	184.0	94.4	187.8	32.1	182.3	35.3	183.0	42.4	186.5	178.7
ADBC	187.0	94.0	188.2	31.5	185.3	33.3	184.7	41.8	185.0	179.1
ADCB	187.2	94.9	188.2	32.0	184.8	33.7	182.7	42.2	186.5	182.2
BACD	172.5	70.8	173.8	32.4	174.7	32.5	160.8	31.7	176.3	164.8
BADC	173.5	71.6	174.7	32.0	174.7	32.4	161.3	31.4	178.3	163.9
BCAD	163.7	133.0	168.2	38.9	164.5	42.3	153.3	41.9	164.2	210.4
BCDA	165.2	235.2	180.3	87.0	161.5	71.1	159.3	85.0	169.3	396.0
BDAC	176.0	407.8	194.3	90.0	169.2	96.8	171.3	143.9	174.0	675.3
BDCA	174.2	412.8	184.8	93.7	161.8	102.7	162.2	146.0	169.5	681.4
CABD	170.2	146.4	175.7	44.9	163.2	43.3	161.0	49.3	169.3	243.5
CADB	171.0	146.5	176.2	45.0	164.3	42.8	161.7	48.6	169.3	243.1
CBAD	172.0	143.7	173.3	47.7	161.7	44.4	167.2	49.6	168.3	235.3
CBDA	173.8	222.2	187.5	77.8	159.3	66.0	170.8	85.7	171.0	402.6
CDAB	174.2	265.3	189.5	73.6	163.5	65.8	165.3	87.0	172.8	448.5
CDBA	172.8	264.5	176.7	73.9	161.3	66.6	161.5	87.0	171.5	446.2
DABC	186.0	375.8	206.5	84.4	177.3	103.5	176.0	148.2	181.8	975.1
DACB	181.8	377.9	207.3	82.9	176.0	103.1	176.0	149.6	181.7	979.1
DBAC	178.3	376.2	200.5	86.6	169.0	103.4	172.2	147.4	179.3	968.2
DBCA	173.0	384.0	197.5	87.1	161.3	106.9	168.5	150.2	173.5	977.3
DCAB	177.5	386.8	209.7	88.0	168.8	107.0	177.7	153.5	174.8	985.4
DCBA	172.8	386.1	202.7	87.9	163.8	106.7	163.3	154.3	174.2	978.6

ety (2000). （白石洋一訳: 組合せ最適化アルゴリズムの最新手法 - 基礎から工学応用まで, 丸善 (2002)）.
2) 田岡智志，渡邉敏正：AllConf：カンファレンス運営支援 Web システム，*IEICE Fundamentals Review*, Vol.2, No.1, pp.66–80 (2008).
3) 森　靖之：ソフトコンピューティングによる時間割編成の自動化手法，高松大学紀要, Vol.40, pp.141–152 (2003-09-28).
4) 西村直史，徳永　亮，大田直樹，櫻井祐子，岩崎　敦，横尾　真：人工知能学会全国大会プログラム自動作成ツールの開発，人工知能学会全国大会 (2008).
5) 西村直史，大田直樹，櫻井祐子，岩崎　敦，横尾　真：制約充足／最適化テクニックを用いた会議プログラム自動作成ツール，人工知能学会全国大会 (2009).
6) 松尾　豊，友部博教，橋田浩一，中島秀之，石塚　満：Web 上の情報からの人間関係ネットワークの抽出，人工知能学会論文誌，Vol.20, No.1, pp.46–56 (2005).
7) 浅田洋平，松尾　豊，石塚　満：Web からの人間関係ネットワークの抽出，人工知能学会論文誌，Vol.20, No.6, pp.370–378 (2005).
8) 畑　守之，田岡智志，渡邉敏正：カンファレンスプログラムの自動生成について，第 23 回 回路とシステム軽井沢ワークショップ，pp.215–220 (2010).
9) 柳浦睦憲，茨木俊秀：組合せ最適化 - メタ戦略を中心として (経営科学のニューフロンティア)，朝倉書店 (2001).
10) Aarts, E. and Korst, J.: *Simulated Annealing and Boltzmann Machines: A Stochastic Approach to Combinatorial Optimization and Neural Computing*, Wiley (1989).