情報処理学会論文誌　数理モデル化と応用　Vol. 3　No. 2　99–107 (Mar. 2010)

集合被覆法を用いた船舶スケジューリングにおける複数オーダの同時積み付け計画の立案について

瀬　田　剛　広[†1]

船舶による輸送計画を立てるスケジューリング問題，特に複数オーダの同時積み付け計画を含めた問題に対する，既存手法の定式化およびアルゴリズムの拡張を提案した．この拡張は複数オーダの同時積み付け計画の考慮範囲を広げ，最適化を行うことを可能にするものであり，この拡張を用いることによってより最適化された計画が得られる問題が存在することが実験例により示された．また，この拡張は全体の計算速度を大きく落とすものではなく，計算時間に合わせてパラメータを調整し，既存手法の計算速度を維持することが可能な手法である．

Way of Order Pairing in Set-cover Ship-scheduling Algorithm

Takahiro Seta[†1]

A new formulation and a corresponding algorithm for ship scheduling problems with multiple order pairing is suggested. The suggested method makes it possible to optimize schedules using wider pattern of pairing way, and it is shown that there exists an example which can be solved with better solution by the method. In addition, the suggested method has a parameter with which the calculation time can be tuned, and does not take much longer time than the existing methods.

†1 独立行政法人海上技術安全研究所
National Maritime Research Institute, Japan

1. はじめに

OR の分野でさかんに研究されてきた，そして今でも研究されている問題として配送計画問題（Vehicle Routing Problem，VRP）がある．この問題は主に陸上のトラックでの輸送を扱う問題であるが，海の世界でも陸と同様，船舶スケジューリング問題（配船計画問題）が存在する．しかし，その研究は陸における配送計画問題ほど進んでいるとはいい難い．特にわが国は周囲を海に囲まれ，造船などにおける一流の技術を持ちながら，この分野の研究の事例は非常に少ない（配船分野の研究動向については，サーベイ論文 1)，5)，6) が存在し，分かりやすくまとめられている．いくつかの例について，比較的詳細にまとめられた，日本語の文献 3) も存在する）．

環境問題を考えても，効率的な船舶の運航が行われることは，単に運航効率の面だけでなく，モーダルシフトの推進による環境負荷減少という二重の効果が期待され，社会的意義も大きい．また，いくつかの船会社の配船の様子を見ても，人手に頼る部分が非常に多く，近年の技術革新の反映余地が大きい分野である．

陸上における VRP において様々な問題設定が存在するように，配船計画においても様々な問題設定が存在する．本稿では特にタンカーを対象とする，複数オーダの同時積み付けを考慮した配船計画問題を扱う．VRP が扱う陸上における輸送では複数の荷物の混載を考えるのは通常のケースであるが，タンカーの配船計画においては，トラックと船舶という輸送体の違いとともに，輸送品が液体であることなどによる異なる性質があり，トラックの VRP とは分けて扱うべき問題となっている．

この問題に対する既存研究として，坂口ら[7] による制約プログラミングを用いた解法や久保・小林[4] による集合被覆法による解法などが提案されている．著者は，特に複数オーダの同時積み付け方法の決定に関し，久保・小林[4] の定式化の変更を提案した[8]．これは，高速性を維持しつつ，複数オーダの同時積み付け法の考慮範囲を広げることを可能にするものである．本稿は，その手法について改めて整理・改良してまとめるとともに，その有効性を確認する実験報告をする．

2. 問題の概要

本章では，本稿で扱う問題の概要を述べる．

2.1 用語の定義

本稿において用いる用語として，特に説明が必要と思われるものとしては以下のものが

© 2010 Information Processing Society of Japan

ある．

積・揚　船舶での輸送においては，陸から船に荷物を乗せることを積荷役と呼び，船から陸に荷物を降ろすことを揚荷役と呼ぶ．同様に荷役を行う場所をそれぞれ積地（積港），揚地（揚港），荷役を行う日をそれぞれ積日，揚日と呼ぶ．

オーダ　荷物の輸送依頼をオーダと呼ぶ．本稿で取り扱う問題の入力として与えられるものである．1 つのオーダは 1 隻の船舶で運ばれなければならないという意味で不可分である．本稿で扱った実験ではそのようなケースはないが，1 つのオーダが複数の荷物を含んでもかまわない．複数の荷物をまとめて運ぶことが指定された場合は，積地や揚地が一致しているかどうかにかかわらず，定式化上は 1 つのオーダとして扱う．

タスク　船舶に同時に積み付けられるオーダの集合，すなわち，船舶が完全に空の状態から，次に空になるまでに運ばれるオーダの集合を久保・小林[4] にならい，タスクと呼ぶ．このタスク，すなわちどのようにオーダを組み合わせるかを求めながら，船にオーダを割り当て計画を立てるのが，本稿で扱う問題である．タスクに含まれるオーダの数には制限は設けず，1 以上の任意の個数のオーダを組み合わせることが可能である．

タスク（あるいはタスクの候補）を作成する問題は，久保・小林[4] にならいペアリング問題と呼ぶ．

2.2　問題の概要

船舶スケジューリング問題，なかでも特に配船計画問題とは，船会社の支配下の複数の船舶（船隊と呼ぶ）の指定された期間の運航計画を立てるものである．対象とする船舶や期間などでいくつかに分類される[1] が，なかでも本稿で扱う事例は「内航」の「不定期船」を対象とし，「オーダ式」のものである．「内航」は国内の港間の輸送を対象にする．国外との輸送を扱う「外航」に比べ，輸送が頻繁に行われるため，配船作業の頻度が多いという特徴を持つ．「不定期船」は，その時点での輸送需要に応じて船の割当てを定める問題であり，短期的な意志決定を高速に行うことが重要になる．対する「定期船」は，船ごとに決まった航路およびスケジュールに従って繰り返し長期間運航を行うモデルで，運航ダイヤや航路ネットワークなどの決定を長期的な視点から行う問題になる．最後の「オーダ式」は「不定期船」のモデルの 1 つであり，荷主から，「どのような荷物（What）を，どれくらいの量（How much），どこからどこまで（Where），いつからいつまで（When）に運んでほしい」という要望（オーダ）を受け，それに従って支配下の船舶の運航計画を定めるものである．VRP の枠組みでいえば Time Window があり，Pickup と Delivery とがある問題に相当する．船舶は広域を動き，航海は複数日にまたがるため，長距離トラックの問題に近いものとなっている．「オーダ式」でない枠組みとしては陸上における Inventory Routing Problem にあたる，在庫を見ながら配船する問題が存在する．

オーダ式の配船においては，オーダをすべて処理しきれない場合に他の船会社などから船の融通を受けて処理したり，逆に支配下船舶が余っている場合に，他社に貸し出したり，本来の輸送対象の荷物以外のものを運んだりというようなことがある．本稿の実験で扱った事例においては，船を他社に貸し出すという例はなかったが，船が足りない場合には，一時的に船を借りて荷物を運ぶ「スポット傭船」と呼ばれる枠組みを利用していた．ただし，スポット傭船は支配下船舶に比べ割高であるため，できる限り使わないというのが船会社の第 1 の要望であった．また，入力として，同時に輸送する荷物の組合せが荷主あるいは配船担当者によって完全に指定される設定もあれば，同時に輸送する荷物を決定する部分まで計画対象とする設定もあるが，本稿では後者の問題設定を扱い，組合せは指定されずアルゴリズムの側で決定する．

船舶の問題を扱うにあたって，特に注意すべき点としては以下の点があげられる．まず，船舶のトラックと大きく異なる特徴として，設計を含めて注文生産で造られ，基本的には船隊内の船それぞれに性能が異なるということがある．そのため，計画時は船舶を 1 隻 1 隻区別して扱うことが求められる．陸上でも積載量の異なる複数の車輌を考慮することはあるが，船舶の場合，一般的にはさらに，荷役速度，移動速度なども異なるという特徴がある．また，上にあげたスポット傭船などという形があるものの，スポット傭船は高額であるだけでなく，要求に見合った性能や特徴の船があるとは限らない（港に着いた後に船体の高さの問題が判明した例などもある）など，需要に応じて船舶を追加・削減するということは難しいという特徴もある．さらに，船舶の運航には気象・海象の影響による遅延・早着が多発するという点があげられる．そのため，人間が配船を行いタイムスケジュールを計画する場合は，細かい例でも数時間単位や 1 日を午前と午後と 2 つに分けるといった程度で，陸上での計画に比べて粗い指定となる．同時に，1 度立てた計画も頻繁に更新することが求められ，アルゴリズムの実行は分単位であることが望まれる．計画期間としては数日から長くて 1 カ月程度が対象となる．これは，内航海運の場合，港から港までの航海に数日を要するため，船舶 1 隻あたり最大で 10 タスク程度を運ぶ期間に相当する．また，陸上の VRP では主に時間の制約要因となるのは移動時間と考えられるが，船舶の場合は輸送量が大きいこともあり，荷役にかかる時間も大きな割合を占める．半日程度から大型船では日をまたがって荷役を行うこともある．

次に，本稿の実験で扱った問題設定について述べる．ただし，本稿の提案部分である配船

© 2010 Information Processing Society of Japan

計画の定式化については，「オーダ式で複数オーダの同時積み付けを考慮する」という問題設定が合えば様々な問題に適用可能である．本稿では，ある船会社の実際の業務および船の運航実績をもとに問題設定およびテストデータを作成した．運航実績で同じ船舶で輸送された複数の製品については 1 つ 1 つ分離し，それぞれをオーダとしたテストデータを作成し，この分離されたオーダの同時積み付け法（ペアリング方法）の計算も問題の一部として問題設定を行った．船舶の有効利用のため複数のオーダをまとめることも含めて，配船担当者の業務の一部であることからこの設定とした．ただし，記録に残っていないものの，現実の配船では，積地・揚地・積日・揚日が一致し，かつ同じ船舶で運ばれているもののいくつかは，同時に輸送することが定められていたものと考えられ，テストデータは実際よりややオーダを細かく分割しすぎている恐れは存在する．実際の運航においては船が傾くことがないよう，荷の積み付け位置にも注意を払うが，その考慮は除外した．オーダが要求する積日・揚日については，実績の積日・揚日が要求されたものとしてテストデータを作成した．積日については指定された日に行うものとし，揚日については指定日以前に到着すればよいものとした．実際の運航においては，まったく同じ時間に 2 隻の船が同時に同じ場所で荷役を行うことは許されないが，本稿の問題設定では考慮不要とした．これは，現実世界においても，荷役設備の利用順序の調整は直前になって行うケースが多いようであり，また，積日・揚日を現実に合わせ指定したことから，実績で競合しなかった荷役が配船結果で競合することは少ないと考えられるためである．最適化の目的関数は，支配下船舶でできるだけ多くの荷物を運ぶという条件のもと，期間内の船舶の総運航距離の最小化することとした．

なお，本稿の実験においては特に内航タンカー（ケミカルタンカー）を対象としている．タンカーは，

- 船体内に「ホールド」と呼ばれる区画（タンク）を複数持つ．なお，本稿の実験で扱った事例の場合，1 隻あたりのホールド数は 4 または 5 区画である，
- 船の形状が単純な直方体ではないということもあり，同じ船舶のホールドであっても互いに容量が等しいとは限らない，
- 輸送品は液体または粉体（ケミカルタンカーの場合は揮発性の油などである）であり 1 度混ざると分離できないため，異なる品種を同じホールドには積載することはできない，

といった特徴を持ち，特に複数オーダの同時積み付けが問題になりやすい性質がある．

3. 定式化とアルゴリズム

本稿の目的は久保・小林[4] の定式化を改良することにより，当該論文中のルーティング問題の定式化上の制約により必ずしも考慮しきれていなかったペアリングの考慮幅を広げ，より良いアルゴリズムに拡張することにある．そのため，ここでまず，久保・小林[4] の定式化を確認し，その後，本稿でどの部分を変更し，アルゴリズムを構成するのかを示すこととする．

3.1 既存手法の概要および問題点

本稿が改良するアルゴリズムの流れは以下のとおりである．

アルゴリズムの全体の構成は，対象とする配船問題をペアリング問題とルーティング問題との 2 段階に分けて解くものとなっている．ここでペアリング問題とは，用語の定義で述べたとおり，タスク，すなわち複数のオーダを同時に運ぶ組合せを求める問題であり，ルーティング問題とは，求めたタスクをそれぞれ支配下船舶に順序も含めて割り当て，各船舶の運航計画（ルート）を定める問題である．ペアリング問題，ルーティング問題はそれぞれ，内部でさらに主問題と子問題との 2 段階（列生成法を用いることからの名称である）に分かれる形となっている．子問題は可能なタスクまたはルートを列挙する部分であり，主問題はその中から，必要なもののみを選択する部分である．ペアリング問題の場合，子問題はタスク候補を列挙する問題であり，主問題はその中からルーティング問題において利用するタスクの集合を決定する部分である．ルーティング問題の場合，子問題は各船舶ごとの可能なルート（タスクの列）を列挙する問題であり，主問題は列挙されたルートから各船舶に 1 つずつ選んで割り当て，最終的な計画を計算する問題である．

ルーティング問題の主問題の定式化は以下のとおりである．なお，変数名および添字については本稿の表記に従い変更している．

$$\text{min.} \quad \sum_{v \in V} \sum_{r \in R_v} C_{vr} x_{vr} + \sum_{t \in T} F_t y_t \tag{1a}$$

$$\text{s.t.} \quad \sum_{(v,r):t \in T_{vr}} x_{vr} + y_t = 1 \quad \forall t \in T \tag{1b}$$

$$\sum_{r \in R_v} x_{vr} \leq 1 \quad \forall v \in V \tag{1c}$$

$$x_{vr} \in \{0,1\} \quad \forall v \in V,\ r \in R_v \tag{1d}$$

$$y_t \in \{0,1\} \quad \forall t \in T \tag{1e}$$

V は支配下船舶の集合を，R_v は船舶 v のルート候補の集合を，C_{vr} は船舶 v がルート r に従って運航した場合のコストを，T はタスクの集合を，F_t はタスク t をスポット傭船で運ぶ場合のコストを，T_{vr} は船舶 v がルート r で処理するタスクの集合を表し，これらが問題

© 2010 Information Processing Society of Japan

への入力である．x_{vr} は船舶 v がルート r を選択するかどうか，y_t はタスク t をスポット傭船で処理するかどうかを表すバイナリ変数であり，定めるべき値である．選択する場合，処理する場合が 1，そうでない場合が 0 である．なお，F_t はスポット傭船のコストという意味づけをせず，オーダが支配下船舶で処理されなかった場合のペナルティや逸失利益などと考えてもかまわない．目的関数 (1a) は各支配下船舶 v が選択したルート r のコスト C_{vr} の和に，支配下船舶群では処理できなかったタスクに対して用いたスポット傭船のコスト F_t の和を加えた総コストを最小化することを表す．一般にスポット傭船にかかるコストが大きいことから，F_t は C_r と比較して大きな値となり，支配下船舶のみでできるだけ多くのタスク（オーダ）を処理するように計算することになる．1 つ目の制約式 (1b) は，すべてのタスク t が，支配下船舶がそれを含む経路 r を選ぶことによって，あるいは，スポット傭船が使われることにによって処理されることを表す．2 つ目の制約式 (1c) は，それぞれの船舶 v が選べる経路 r はたかだか 1 つということを表す．全体としてはすべてのタスクの集合をその部分集合であるルートで分割（被覆）する集合分割（被覆）問題の形式となっている．

このルーティング問題に対する子問題は十分な要素数を持つ R_v を生成することに相当する．この生成は，連続して実行可能なタスクと開始時刻とを表現した時空間ネットワーク上で最短路問題（あるいはパス列挙問題）を解くことによるが，本稿の目的から外れるため，詳細は割愛する．

次にペアリング問題の主問題の定式化は以下のとおりである．ルーティング問題と同様，変数名および添字については本稿の表記に従い変更している．

$$\text{min.} \quad \sum_{t \in T} C_t x_t + \sum_{o \in O} F_o y_o \tag{2a}$$

$$\text{s.t.} \quad \sum_{t: o \in O_t} x_t + y_o = 1 \quad \forall o \in O \tag{2b}$$

$$x_t \in \{0,1\} \quad \forall t \in T,\ y_o \in \{0,1\} \quad \forall o \in O \tag{2c}$$

T は可能なタスクの候補の集合，C_t はタスク候補 t のコスト，O はオーダの集合，F_o はオーダ o を単独で運ぶ場合のコスト，O_t はタスク候補 t に含まれるオーダの集合であり，これらが問題への入力である．x_t はタスク候補 t を採用するかどうか，y_o は 1 つのオーダ o のみからなるタスクを採用するかどうかを表し，定めるべき値である．目的関数 (2a) は選ばれたタスク全体のコストの総和を表し，制約式 (2b) は各オーダに対し，それを含むタスクを 1 つに限定することを表す．ルーティング問題同様，すべてのオーダの集合をその部分集合であるタスクで分割（被覆）する集合分割（被覆）問題の形式となっている．なお上の定式化において，タスクの候補の集合に対しては，T というルーティング問題におけるタスクの集合と同じ文字を用いたが，ペアリング問題において x_t あるいは y_o が 1 となったタスクがルーティング問題の T として残されるものであり，一般にこれらは互いに異なる集合である．

ペアリング問題に対する子問題は十分な要素数の T を生成することに相当する．既存手法においてはこれを混合整数計画問題として定式化し，列生成法を用いて求めているが，本稿の目的から外れ，また，本稿で扱った事例とは定式化が一致しなかったため，詳細は割愛する．

この手法の優れた点としては，全体の問題をサイズの小さな問題に分けて解くことにより，高速に計画を立案することが可能である点があげられる．Kim・Lee[2] が述べているように，集合被覆法のメリットとしては，複雑な制約条件の反映を最終的な計画立案（集合被覆段階）の前段階（ルート生成段階）で行うことで，数式で表しにくい制約条件であっても比較的容易に反映させることができ，かつ高速化も行えるという点がある．さらにこの久保・小林[4] の手法は，同様な枠組みをルート生成段階に再度適用し，ルート生成をルーティング問題とその前段階のペアリング問題に分離することにより，柔軟性の向上と高速化とを達成したものといえる．集合被覆法は列生成法と組み合わせることが多いが，ペアリング問題を分離することにより，列生成法において，時空間ネットワーク上での最短路問題という扱いやすい問題で済むようにし，列生成の反復の高速化に成功している．

以上が既存手法の概要であるが，本稿では，既存手法がペアリング問題において，各オーダに対しそれを運ぶタスクを 1 つに制限している，という点を改善すべきポイントとして着眼した．この制限が妥当性を持つためには後に続くルーティング問題の要請に対して望ましいタスクを選択できている必要がある．しかし，既存手法におけるタスクの選択法は最適性が保証されているわけではないため，事例によっては，望ましいタスクを選ぶことができないものと考えられる．特に，一般に船舶は 1 隻ごとにそれぞれ異なることから，船舶によっては同時に運ぶことができないオーダの組合せがあるため，ルーティング問題を解く前の段階でオーダの組み合わせ方を 1 通りに限定する定式化では良い計画が得られない恐れがある．また，この制限はルーティング問題の定式化からきているため，ペアリング問題だけでなく，ルーティング問題の定式化も含めて見直す必要がある．

なお，実際に上記のような既存手法に対する問題が発生する例として，人工的な問題であれば，以下のような問題例が構成可能である．

船隊構成　2 オーダまで同時に運べる大型船舶 1 隻と 1 オーダまでの小型船舶 2 隻

© 2010 Information Processing Society of Japan

オーダ　積地・揚地・積日・揚日・荷物ともまったく同じ内容の 4 つのオーダ

この場合，期待される最適な計画は，4 オーダを 2 + 1 + 1 に分け，3 隻の船舶に運ばせるものであるが，既存手法を適用した場合，ペアリング問題において，4 オーダを 2 + 2 に分けてしまい，ルーティング問題は失敗してしまう．

3.2　提案する定式化と対応するアルゴリズム

本節では，本稿において提案する定式化を述べる．なお，既存手法との差異は，式中において太字で強調している．あわせて定式化の変更にともない可能となる，あるいは要求されるアルゴリズムの修正についても述べる．

3.2.1　ルーティング問題

まず，ルーティング問題の定式化 (1) を以下のように変更する．

$$\text{min.} \quad \sum_{v\in V}\sum_{r\in R} C_{vr}x_{vr} + \sum_{t\in T} F_t y_t \tag{3a}$$

$$\text{s.t.} \quad \sum_{(v,r):\boldsymbol{o}\in \boldsymbol{O_{vr}}} x_{vr} + \sum_{\boldsymbol{t}:\boldsymbol{o}\in \boldsymbol{O_t}} \boldsymbol{y_t} \geq 1 \quad \forall \boldsymbol{o}\in \boldsymbol{O} \tag{3b}$$

$$\sum_{r\in R_v} x_{vr} \leq 1 \quad \forall v\in V \tag{3c}$$

$$x_{vr}\in\{0,1\} \quad \forall v\in V,\ r\in R_v \tag{3d}$$

$$y_t\in\{0,1\} \quad \forall t\in T \tag{3e}$$

O_{vr} は船舶 v のルート r に含まれるオーダの集合，O_t はタスク t に含まれるオーダの集合を表し，その他の変数の定義は変わらない．

既存手法[4] との違いは 1 番目の制約条件 (3b) をオーダ単位で立式している点である．これにより，オーダ単位でスポット傭船の利用を判断することができるようになり，1 つのオーダを含むタスクを複数用意することが可能となる．また，同じ制約式において，= ではなく ≥ を用い，集合分割ではなく，集合被覆として立式している．自然な定式化は集合分割としての定式化であるが，この後提案するペアリング問題の定式化の変更により，集合被覆は可能でも集合分割は不可能となる場合があるため，集合被覆による立式を用いるものである．もし，同じオーダが二重に処理されてしまう結果が出た場合は，その段階で該当オーダを含むタスクからそれを取り除くことで望ましい集合分割の結果が得られることになる．

また，この定式化の修正は，既存手法に対し互換性を持つものである．既存手法同様，オーダを含むタスクの数を 1 つに制限した場合，完全に既存手法と同じ問題となる．この際，まったく同じ形の制約式が複数現れることにはなるが，通常の MIP ソルバであれば前処理で余分な制約式は除去されるため，計算速度に与える影響も軽微と期待される．

さらに，式のうえには現れないが，まったく同じオーダの組合せのタスクを複数用意することも可能である．実際，本稿の実験における実装では，港を回る順序によって，移動にかかる距離や実行可能性が変わることから，タスクにその情報も含め，まったく同じオーダの組合せであっても，港を回る順序が違うものは異なるタスクとして扱うこととしている．

なお，久保・小林[4] のようにルーティング問題の子問題を解く際に列生成を用いる場合は，アルゴリズムに修正が必要である．定式化において制約式をオーダ単位で立式するよう修正したことから，対応する双対変数値の意味づけが，「タスクを処理する価値」ではなく，「オーダを処理する価値」を表すものに変化している．よって，列生成においてタスクに対する双対変数値を利用する部分は，タスクに含まれるオーダすべてに対する双対変数値を合計したものを用いるように変更する必要がある．

また，上では T のすべてのタスクについて y_t および F_t を用意しているが，一部のタスクのみについて y_t および F_t を用意する形でもよく，逆に T に含まれないタスクに対して y_t および F_t を用意してもかまわない．一般に支配下船舶で扱えるタスクと 1 回のスポット傭船によって扱うことができるタスクとが一致する保証はないが，これによりそのような事例にも対応可能である．特に，オーダ 1 つからなるタスク t については必ず y_t および F_t を用意することとすれば，実行可能解の存在を保証することができる．あるいは，その特殊な形として，スポット傭船に対応する変数 y や F をタスクに対してではなくオーダに対して用意した下の形式を用いてもよい．オーダごとに輸送料金が定まっていて逸失利益として表現したい場合や，単に支配下船で運べなかったオーダにペナルティをかけたい場合など，事例によってはこちらの形がより状況を正確に表すことが可能と考えられる．本稿の実験における実装も，この方式を用いている．

$$\text{min.} \quad \sum_{v\in V}\sum_{r\in R} C_{vr}x_{vr} + \sum_{\boldsymbol{o}\in \boldsymbol{O}} \boldsymbol{F_o}\boldsymbol{y_o} \tag{4a}$$

$$\text{s.t.} \quad \sum_{(v,r):\boldsymbol{o}\in \boldsymbol{O_{vr}}} x_{vr} + \boldsymbol{y_o} = 1 \quad \forall \boldsymbol{o}\in \boldsymbol{O} \tag{4b}$$

$$\sum_{r\in R_v} x_{vr} \leq 1 \tag{4c}$$

$$x_{vr}\in\{0,1\} \quad \forall v\in V,\ r\in R_v \tag{4d}$$

$$\boldsymbol{y_o}\in\boldsymbol{\{0,1\}} \quad \forall \boldsymbol{o}\in \boldsymbol{O} \tag{4e}$$

ここで，F_o はオーダ o をスポット傭船で処理する場合のコスト，y_o はオーダ o をスポット

© 2010 Information Processing Society of Japan

傭船で処理するかどうかを表すバイナリ変数である．

3.2.2　ペアリング問題

次に，ペアリング問題の定式化を変更する．すでに述べたルーティング問題の定式化の修正により，ペアリング問題の出力として，オーダごとのタスクの数を 1 つに限定しなければならないという制限が消えている．したがって，既存手法の制約式 (2b) を緩めれば，目的を達成することができる．

たとえば，1 つのオーダに対し，それを含むタスクを N 個程度残すなら，既存手法において，各オーダに対して，それを含む効率的なタスクを求める定式化を以下のように修正する．

$$\text{min.} \quad \sum_{t \in T} C_t x_t + \sum_{o \in O} F_o y_o \tag{5a}$$

$$\text{s.t.} \quad \sum_{t: o \in O_t} x_t + y_o \ \geq \boldsymbol{N} \quad \forall o \in O \tag{5b}$$

$$x_t \in \{0, 1\} \quad \forall p \in P,\ \boldsymbol{0} \leq \boldsymbol{y_o} \leq \boldsymbol{N} \quad \forall o \in O \tag{5c}$$

これにより，各オーダ o に対し，それを含むタスクを N 個程度残すことが可能となる．ここで，正確に N 個である必要はないため，式 (5b) においては，$\geq$ を用い N 個以上を可能とした．また，すべての可能なタスクを生成しても N 個のタスクが見つからないような場合にも実行可能解の存在を保証するため，式 (5c) において，y_o の範囲を広げ，N 個以下も可能とした．このように条件を緩めても，目的関数の最小化の条件があるため，無制限に大きくなることも小さくなることもなく，N 個程度残されることとなる．

上の式では既存手法との変更部分を最小限にするため，変数名などは変えなかったが，オーダ 1 つだけからなるタスクを特別扱いせず，それらを T に含めてしまい，y_o および F_o の解釈を変え，単なるソフト制約化のための変数および制約違反のペナルティとして扱うことができる．この場合，F_o は純粋にペナルティを表すため，オーダを単体で運ぶ場合のコストとは無関係に大きな値を用いることが可能となり，多くのタスクが選択されることが期待される．また，上では整数計画問題として定式化しているが，オーダあたりのタスクの数は任意であり N 個にこだわる必然性もないことから，整数条件を外し，x_t を $[0, 1]$ をとる連続的変数として，ゼロ以外の値を持ったものについては残す，というルールを用いることも可能である．多くのケースでは整数条件を外しても 0 以外の値を持つ変数の数が無制限に増えることはないと期待されるが，場合によっては残す個数の上限を定めてもかまわない．この方法を用いれば，ペアリング問題の主問題に時間がかかるような問題例に対しては高速化が期待される．

さらに，提案手法のペアリング問題において，オーダごとのタスクの数は共通である必要はない．一般的な傾向としては，より多くのタスク候補を生成することは後に続くルーティング問題における計画結果の品質向上に資すると考えられる．そのため，オーダ単位のタスク生成数を共通の N ではなく，オーダごとに定まる N_o とし，

$$\text{min.} \quad \sum_{t \in T} C_t x_t + \sum_{o \in O} F_o y_o \tag{6a}$$

$$\text{s.t.} \quad \sum_{t: o \in O_t} x_t + y_o \ \geq \boldsymbol{N_o} \quad \forall o \in O \tag{6b}$$

$$x_t \in \{0, 1\} \quad \forall t \in T,\ \boldsymbol{0} \leq \boldsymbol{y_o} \leq \boldsymbol{N_o} \quad \forall o \in O \tag{6c}$$

という形で立式し，特に重要なオーダなどに大きめの N_o を設定するといったことも可能である．

また，問題のサイズが十分に小さい場合，$N = \infty$ として T の中から選別を行わず，生成，列挙されたタスク T すべてをルーティング問題に引き渡すことも可能である．これが可能になったのもルーティング問題の定式化を修正したことによる．

その他，3.1 節で示した人工的な例のように，船舶の性能がそれぞれ違うことから，全船隊一括でペアリング問題を解くと，小型の船舶など制限の大きい船舶が扱えるタスクが残されなくなることがありうるため，船舶ごと個別にペアリング問題を解き，その結果を統合してルーティング問題へ引き渡すという手法なども可能である．

なお，本稿の提案手法は，ペアリング問題の子問題を解くための手法については指定しない．後に示す実験においては実行可能なすべてのタスクを生成しているが，事例によっては，ペアリング問題を解く際，久保・小林[4] の列生成の手法を利用することで，高速に計算を行える可能性が存在する．提案手法の定式化に合わせて列生成法を再構成することも可能と考えられるが，列生成に用いる双対変数値を得るための式は提案手法ではなく既存手法の式を用い，最終的なタスクの選択にのみ提案手法の定式化を用いれば，既存手法で列生成法が利用できた事例については，そのまま列生成法が利用でき，ペアリング問題の計算速度への影響は無視できるものと期待できる．

また，オーダごとのタスクの数 N については問題の大きさや複雑さによって変化させることが望ましいと考えられるが，これについても本提案ではその設定法を指定しない．この N の最適な値を選ぶのは難しく，理論的に定まるものではないと考えられる．定式化 (5) のような単純な手法を用いた場合，3.1 節で示した人工例であれば $N = 5$ で最適解が求ま

© 2010 Information Processing Society of Japan

るが，どんな N に対してもそれでは足りないような例が容易に構成可能である（人工例のオーダ数を増やせばよい）．しかし，多くの事例においては，たとえば，単に N を 5 や 10 に固定しても，確率的には十分な数のタスクを残すことが多くなるものと期待されるため，既存手法に比べて安定的に優位性を持った結果を出すものと考えられる．また，N を 5 や 10 にした結果に加えて，N を 1 にした場合の結果との和集合をとることとすれば，既存手法で選ばれていたタスクは残ることが保証され，既存手法に比べた，計算時間以外の性能のさらなる安定的優位性が期待される．そのほかにも，たとえば，式 (6) の形の定式化を用いた場合，ルーティング問題を（最後まで，あるいは部分的に）解き，そのうえで求まるオーダの価値（制約式の双対変数値など）に応じて N_o を変化させて，新しく生成されたタスクを追加するといった形で，列生成法に似た考え方でペアリング問題を子問題とし，ルーティング問題を主問題として繰り返すような反復解法なども考えられる．

4. 実験および結果

本稿の提案手法については，すでに示したように提案手法でなければ求まらない人工的な例が存在するため，一定の優位性があることは示されている．これに加え，現実の事例をもとにしたデータで，既存手法では不十分と思われる事例が存在するかどうか，また，既存手法に対し，速度面での低下がどの程度発生するのかを確認するため実験を行った．本章において，その結果を示す．

実験に利用したテストデータおよび環境は以下のとおりである．テストデータとしてはあるケミカルタンカーの船隊の運航実績をもとにしたデータを利用した．利用したテストデータに含まれる船舶は 7 隻，オーダは 111 件，期間は約 1 カ月（最初の積日から最後の揚日までで 32 日間）であった．船舶 1 隻あたりのホールドの数は前述のとおり最大 5 区画，1 航海あたりの期間は 1 日から 4 日まで幅があるが，平均して 3 日程度，同時に運ぶオーダ数も幅があるが最大 4 つであった．計算環境としては，CPU が Intel Pentium4 3.4 GHz, RAM が 4 GByte である Windows XP マシンに混合整数計画ソルバとして，IBM 社の CPLEX 11.2 を用いた．プログラムは C++言語で作成し，Microsoft 社の Visual Studio 2005 Professional Edition の Release ビルドでコンパイルしたものを用いた．目的関数としては，支配下船舶で処理しきれなかったオーダ数を最小化することを第 1 の目的とし，期間内の総航行距離を最小化することを第 2 の目的とした．

なお，本稿で扱った事例においては，タスクの生成部分に列生成法を適用して高速化できる目途が立たなかったため，タスクの全列挙を行い，その中から有用なタスクのみを絞り込む部分（ペアリング問題の主問題）について，提案手法を用いることとした．これは，問題設定の差があるためと考えられるが，時間に関する制約の存在などにより，本稿の事例では久保・小林[4] に示されたような簡単な子問題の定式化では列生成が行えず，列生成の反復 1 回にかかる計算時間が全列挙を行うのと同程度となってしまったためである．ただし，全列挙とはいっても，どの船舶にとってもメリットのない遠回りの経路などは，除外している．

表 1　実験結果：ペアリング問題におけるパラメータ N を変化させた場合
Table 1　Numerical result: changing parameter N in the pairing problem.

条件	利用タスク総数	オーダ平均タスク数	処理失敗オーダ数	総航行距離	計算時間（秒）
$N \equiv 1$	72	1	0	32,009	5
$N = 1$	72	1	0	32,009	4
$N = 2$	116	1.7	0	31,541	14
$N = 3$	145	2.4	0	31,888	28
$N = 4$	163	2.9	0	31,549	31
$N = 5$	183	3.4	0	31,549	30
$N = 10$	250	5.6	0	31,569	66
$N = 15$	286	7.0	0	31,571	77
$N = \infty$	380	11.0	0	31,342	126
実績値	74	1	1	32,630	—

表 1 は，テストデータに対し，提案手法のペアリング問題におけるオーダ単位で残すタスク数を表すパラメータ N を変化させた実験の結果を表したものである．ルーティング問題においては，式 (4) の定式化を用いた．ペアリング問題においては，式 (5) の定式化を用い，x_t は整数制約を含めたバイナリ変数とした．オーダ 1 つからなるタスクについては特別扱いせず，y_o はペナルティとして扱い，対する F_o としては十分に大きな値を指定した．表中の $N = \infty$ は生成されたタスクをすべてルーティング問題で利用した場合の結果を表す．$N \equiv 1$ は列生成を用いていない点を除き，既存手法[4] と同等の手法を表す（$N = 1$ との違いは式 (5b) において，$\geq$ ではなく $=$ を用いている点であるが，計算結果はまったく同じであった）．実績値は人手による配船結果である．実績値であるため，本来，処理失敗オーダ数は 0 であるはずだが，ヒアリングによって得た条件に従いシミュレーションを行ったところ，1 つのオーダについて間に合わない結果となったため，処理失敗オーダ数が 1 となっている．「利用タスク総数」はペアリング問題が終了した後で残されたタスクの数である．「オーダ平均タスク数」はオーダごとにそれを含むタスクの数を数えて，平均したものであり，可能なタスクの組合せが十分あれば N とほぼ等しくなるべきものである．なお，「オーダ平均タスク数」は，「利用タスク総数」をオーダ数で割った値とは一致しないが，複

© 2010 Information Processing Society of Japan

数オーダからなるタスクは 2 回以上数えられるためである．

結果が示すように，既存手法に対応する $N \equiv 1$ に対し，提案手法で N を増やしたものは全体的に高い改善率を示している．すなわち，事例によってはオーダあたりのタスク数を 1 つに制限することは本来可能な改善を制限してしまう恐れがあるといえ，提案手法の実際に存在する事例に対する有効性が示された．実験はあくまでも例にすぎないが，この規模の問題であれば，30 秒程度と十分に許容できる程度の時間をかけることで，既存手法よりさらに 1.5%程度良好な結果を得ることができており，提案手法を用いるメリットは大きいと考えられる．ただし，N を変化させた中で最も良い目的関数値を出しているのは，$N = \infty$ であるものの，次に良いのは $N = 2$ となっており，$N = 3$ では $N \equiv 1$ には勝るものの $N = 2$ と比較した場合は大きく悪化し，それ以降も $N = 2$ とほぼ同程度ながら上回ることはできていない．すなわち，傾向としては N を増やせば安定的に良い性能を出すと期待されるものの，N を増やしたからといって必ずしも単調に目的関数値が改善するとは限らないことがいえる．これは，ペアリング問題において N を増やすことによってルーティング問題に残されなくなるタスクが存在するためだと思われ，ペアリング問題の部分についてはいまだ改善の余地があることもあわせて示す結果ともなっている．本稿の提案したルーティング問題の定式化の拡張により，ペアリング問題の出力方式の自由度が上がっていることから，全体の問題をペアリング問題とルーティング問題との 2 段階に分ける枠組みは活かしながらもまったく違う方式でペアリング問題を解くことも含めて検討する価値があるといえる．

なお，特に表では示さないが，ペアリング問題において，整数制約を外した場合の実験も行ったところ，結果は計算時間・計算結果とも，上に示した整数制約を含めた場合と変わらないものであった．ペアリング問題に出てくる問題は集合分割（被覆）問題あるいはその一般化問題であり，一般的には難しい問題であるため，整数制約を外すことでその部分の高速化が図れる例もあるものと考えられるが，本稿の実験で扱った例は問題サイズが小さくペアリング問題に時間がかからなかったこともあり，特に効果は得られなかった．

5. ま と め

本稿では，複数オーダの同時積み付けを合わせて行うタイプの配船計画問題に対する集合被覆法を用いた手法の改良案を提案した．

改良案は，既存手法に対する定式化上の変更は小さいがその改良効果は大きく，適用可能な問題範囲を広げ，また，同じ問題に対しても質の良い解を出力することが可能な手法となっている．実際の事例をもとにした実験においては，提案手法によって，既存手法を上回る計画を出力することに成功するなどの有効性も示された．

提案手法は多少の計算速度の低下はあるものの，残すタスクの数を表すパラメータにより計算速度をコントロールすることが可能であり，列挙されたすべてのタスクを用い，時間をかけて良い解を得ることも，タスクを残す数は控えめにして，時間と計算結果のバランスをとることも可能な手法となっている点でも，既存手法の柔軟性を高めることに成功している．

また，提案手法の柔軟性向上にともなう拡張可能性に関しても，ペアリング問題における列生成法利用との相性の検証や，ペアリング問題を子問題，ルーティング問題を主問題とした反復実行の有効性の検証などについては実験による確認までは至っておらず，今後の課題があるものの，可能性のある手法となっている．

今後の課題として特に重要なものとしては，ルーティングの段階に残すタスクの数を増やした結果，逆に計算結果の評価値が悪化することもあることがあり，与えられた問題に対し十分な品質の解を出力するためのパラメータ調整法あるいはそれに代わる対策が示されていない点があげられる．

また，既存手法に対する提案手法の計算速度の低下の主な原因は，ペアリング問題で残すタスクの数が増大することにより，ルーティング問題で時空間ネットワークのグラフのサイズが大きくなり，その構成に時間が増大すること，および，それを用いた列生成において回数および 1 回の反復あたりの時間が増大することにあるが，実験における実装では，問題がそれほど大きくないこともあり，グラフの構成の最適化については必ずしも検討していないことから，その部分の最適化の可能性およびその最適化を行った場合の計算時間についても興味のあるところである．

謝辞　この研究は独立行政法人新エネルギー・産業技術総合開発機構（NEDO）の先導研究開発「内航船の環境調和型運航計画支援システムの研究開発」および同実用化研究開発「内航船の環境調和型運航計画支援システムと陸上交通情報連携に関する実用化研究開発」の一部として行ったものである．

また，船会社の方には貴重なデータ・情報などを提供していただいた．この場を借りて改めて感謝を申し上げる．

さらに，査読者および研究発表会において質問いただいた方には，貴重な意見をいただいた．この場を借りて感謝を申し上げる．

© 2010 Information Processing Society of Japan

参　考　文　献

1) Christiansen, M., Fagerholt, K. and Ronen, D.: Ship Routing and Scheduling: Status and Perspectives, *Transportation Science*, Vol.38, No.1, pp.1–18 (2004).
2) Kim, S.-H. and Lee, K.-K.: An Optimization-based Decision Support System for Ship Scheduling, *Computers and Industrial Engineering*, Vol.33, No.3-4, pp.689–692 (1997).
3) 小林和博，久保幹雄：船舶スケジューリング，第 20 回 RAMP シンポジウム論文集，pp.61–75 (2008).
4) 久保幹雄，小林和博：階層的積木法と列生成法の融合—輸送・船舶スケジューリングを例として，計測と制御，Vol.47, No.6, pp.519–524 (2008).
5) Ronen, D.: Cargo ships routing and scheduling: Survey of models and problems, *European Journal of Operational research*, Vol.12, pp.119–126 (1983).
6) Ronen, D.: Ship scheduling: The last decade, *European Journal of Operational research*, Vol.71, pp.325–333 (1993).
7) 坂口　隆，加納敏幸，福村直登，武内陽子：ホールド割当を含む不定期船配船計画の最適化，スケジューリング学会スケジューリング・シンポジウム 2008 講演論文集，pp.139–144 (2008).
8) 瀬田剛広：複数オーダーの同時積み付けを考慮したケミカルタンカーの船舶スケジューリング，情報処理学会研究報告（2009-MPS-73），pp.181–184 (2009).

(平成 21 年 11 月 19 日受付)
(平成 22 年 1 月 6 日再受付)
(平成 22 年 1 月 13 日採録)

瀬田　剛広（正会員）

1979 年生．2005 年東京大学大学院情報理工学研究科コンピュータ科学専攻修士課程修了．2008 年より（独）海上技術安全研究所研究員．計算機を用いた運航計画・配船計画の改善に関する研究に従事．日本船舶海洋工学会会員．

© 2010 Information Processing Society of Japan