修 士 論 文

自動車運搬船における貨物積載プランニングの席割問題に対する局所探索法

252001054　　竹田陽

名古屋大学大学院情報学研究科

数理情報学専攻

2022年1月

自動車運搬船における貨物積載プランニングの席割問題に対する局所探索法

252001054　　竹田　陽

概 要

ある港から複数の港を経由して自動車を積み, 複数の港で自動車を降ろす自動車運搬船における貨物積載について考える. 自動車運搬船に積む自動車の情報が与えられてから実際に自動車が船に積まれるまでに, 席割と呼ばれる注文 1 つ 1 つを船内に割り当てる作業と, シミュレーションと呼ばれる席割で決まった車両を 1 台ずつ詳細に配置する計画を立てる作業の二種類の作業が行われる. 席割作業では船の階層内を一定間隔の大きさに区切ったホールドと呼ばれるスペースに, 与えられた積載自動車リストのどの自動車を何台割り当てるかを決める. シミュレーション作業では席割作業でホールド毎に割り当てられた自動車を, 自動車の向きや空きスペース，作業効率などを考慮してホールド内への配置を決定する. 本研究では，この席割作業とシミュレーション作業 の全自動化への第一歩として席割作業に着目し, 短時間かつ効率の良い席割を出力することを目標とする.

席割問題は割当問題として定式化することができるが, 特徴的な要件として以下の 2 つがある. 一点目は自動車を船に積むあるいは船から降ろす際に, 人が自動車を運転して所定の位置まで移動するための通路を確保する必要があることである. このとき既に積んだ自動車が，新たに積むもしくは降ろす自動車の進行経路上にあると, 自動車は割り当てられたホールドと船の出入り口間を移動することができない. 二点目は船全体の荷重バランスを考慮しなければならないことである. 例えば船が海上を進む際に積まれた自動車が船の一部の領域に集中していると, 船が波に揺られたときにそのまま横転してしまう可能性があり非常に危険である.

良い席割の指標について, 海運会社の業務として席割を作成している作業者や自動車を船内のエリアまで運転して運ぶ運転手へのヒアリングを行った．ヒアリングから，積み降ろし時に意図しない自動車を積み下ろすことが起こりにくいこと, 積み降ろしをする港が同じ自動車が船内でまとまって割り当てられていること，運転手が作業をする効率が落ちない程度のスペースが確保されていること, 船のスペースを無駄にしないことに重点を置いていると考えられた. 本研究ではこのような良い席割の指標を考慮した目的関数を設計する.

本研究では，良い席割を作成するための指標や特徴を考慮した数理モデルを提案する．商用の整数計画ソルバー (Gurobi Optimizer) を使用した厳密解法に基づく数理モデルと，局所探索法を用いた近似解法に基づく数理モデルの 2 種類を提案し，計算時間や精度の比較を行った．計算実験の結果，局所探索法を用いたモデルを用いることで，解の精度をあまり落とすことなく計算時間を短縮できることを確認した．また，局所探索における初期解生成に関して，2 種類のアプローチを提案する．一つ目は，整数計画問題として定式化したモデルの線形緩和問題を解き，その解を初期解として利用するというものである．二つ目は，積む港と降ろす港が同じ注文をまとめて一つとみなし，整数計画問題を解き解を初期解として利用するというものである．提案した 2 種類のアプローチと，ランダムに生成した解の 3 種類を局所探索における初期解として探索を行い，局所探索が終了するまでの計算時間と解の精度の比較を行う.

A local search algorithm for the stowage planning problem for pure car carrier ships

252001054 Kiyoshi Takeda

Abstract

We consider the cargo loading of a car carrier ship, for which cars are loaded and unloaded at several ports. Two kinds of operations called "stowage planning" and "simulation" are carried out after the information of cars to be loaded on a car carrier is given. In the stowage planning, they decide how many cars from the given list of cars should be assigned to the spaces called holds, which the spaces are obtained by dividing a deck of the ship into certain intervals. In the simulation work, the cars assigned to each hold in the stowage planning work are determined to be placed in the hold considering the direction of the cars, available space, and work efficiency. In this study, we aim to automate the work of stowage planning by using mathematical optimization techniques to output efficient stowage planning in a short time.

The stowage planning problem can be formulated as an assignment problem, and there are two characteristic requirements. The first is that when loading or unloading a car onto or off a ship, it is necessary to secure a passage for a person to drive the car to a certain position. In this case, if a car already loaded is in the path of a car to be newly loaded or unloaded, the car cannot move between the assigned hold and the entrance of the ship. Secondly, the load balance of the entire ship must be considered. For example, if the loaded cars are concentrated in some areas of the ship as the ship moves through the sea, it is very dangerous that the ship may roll over when it is rocked by waves.

From the interviews, we have confirmed that it is important not to cause unintended cars to be unloaded during loading and unloading, that cars with the same port of loading and unloading are placed close to each other on the ship, that there is enough space for the driver to work efficiently, and that the space on the ship is not wasted. In this study, we design an objective function that takes these indicators of good stowage planning into account.

In this study, we propose a mathematical model that takes into account indicators and features for creating a good stowage planning. Two types of mathematical models are proposed: one is based on an exact solution method using an integer programming solver, and the other is based on an approximate solution method using a local search method, and we compare the computation time and accuracy. As a result of computational experiments, we confirmed that the model using the local search method can reduce the computation time without losing much accuracy of the solution. We also propose two approaches to generate initial solution. The first one is to solve the linear relaxation problem of the model formulated as an integer programming problem and use the solution as the initial solution. The second approach is to consider all orders with the same loading and unloading ports as a single order, solve the integer programming problem, and use the solution as the initial solution. We compare the computation time and the accuracy of the solution by using the proposed two approaches and a randomly generated solution as the initial solution in the local search.

目 次

第1章　はじめに

本研究では．複数の港で荷物を船に積み, 複数の港で降ろす運搬船を考える．一般的には燃料や原料など多岐に渡る積荷を扱うが，本研究では乗用自動車を運搬する船を考える．一般的に自動車運搬船は, 自動車を船の一定間隔で区切られたホールドと呼ばれるスペースにどの自動車を何台割り当てるかを決定する席割作業と呼ばれる工程を経て, その後席割作業で割り当てられた自動車に対して向きや場所を考慮して一台ずつ船内の領域に配置するシミュレーションと呼ばれる作業を行う． 現状自動車を輸送する会社はこの作業を人手で行っているが少なくとも席割作業に 3 時間, シミュレーション作業に 4 時間かかることから, 多大な人件費と時間をかける必要があり直前の追加注文等に対応が出来ないという問題点がある [1]. 本研究では席割作業とシミュレーション作業の 2 つの中で，席割作業の自動化に対して検証した結果を記す.

席割作業に関しては，斉藤，柳田 [5] が自動車船に対する積み付け計画を行うアルゴリズムを提案している．また，斉藤らは同年に席割に関する初期解生成，解の探索に関して特許を取得している [4]. この特許は全ての車両が搬入可能であるための席割を作成するためのシステムであり，本研究で扱う複数の目的関数などを考慮していない.

本研究で扱う席割作業の概要について述べる．様々な種類の車を A 港から B 港まで輸送するというような積載自動車の注文リストを受け取る．注文リストを受け取ったプランナーと呼ばれる席割を作成する作業者は好ましい席割になるように, 注文の船内スペースへの割当を考える．席割作成における前提条件について説明する．例えば船内の特定の領域に積まれている自動車よりも奥に積まれている自動車が, ある C 港で降ろされて空きスペースが発生したとする．この場合には，次の D 港で積む自動車があればその自動車の積みやすさのために, 船内に積んだ自動車を奥側に詰めて手前の領域を確保するというような既に積まれた自動車の航海中の移動は原則しない．また自動車は全てのホールドに容易にアクセスすることは出来ない．例えば本稿の実験で扱う自動車運搬船は 12 階構造の内 5 階のみに外部から自動車を入れるスロープがあり, 船内の奥側ホールドにアクセスしたい場合はそのホールドにアクセスする際に通過するホールドに十分な空きスペースがないとアクセスすることが出来ない.

本稿では第 2 章で席割問題に対する詳細な問題設定や, 本研究で扱う自動車輸送航海における専門用語について定義する．第 3 章では注文に含まれる自動車一つ一つをどの領域に割り当てるかを最適化する数理モデルと, そのモデルに登場する本研究独自の制約や好ましい席割作成のための目的関数を説明する．第 4 章では提案した数理モデルと商用ソルバーを用いて, 実際の過去の航海データを基に求解実験を行い，比較実験を行う.

第2章　問題設定

複数の港を経由し, 自動車を輸送する運搬船を想定する. 例えば乗用車 100 台を港 A から港 B, トラック 30 台を港 C から港 D というような各注文を, 運搬船の階層毎に一定の広さで区切られたスペースへの割当を計画する. この作業を席割作業 (stowage plan)[3] という. 本稿では複数の種類がある運搬船のなかで，単純な船の構造データを用いて実験する. また実際の席割業務では積載する自動車の種類が多様であり, ブルードーザーやショベルカーなどの建機は高さや重さが乗用車とは大きく異なるので特定の領域にしか収容出来ないという問題がある. これに対し本稿では作業自動化の初期段階として，自動車を全て乗用車とし, 船内の任意領域においてどの自動車も積載が可能と言う条件で席割を作成する. この章では一般的な船への自動車積載における専門用語の定義, 席割作業における入力情報や出力情報の説明をする.

2.1　用語定義

本研究で扱う船の航海等に関する専門用語の定義をする.

- 席割
  注文一つ一つを船のホールドに割り当てる作業.
- シミュレーション
  席割で決まった自動車を一台ずつホールド内の領域に貼り付ける作業.
- プランナー
  席割やシミュレーションを考える作業者.
- オペレーター
  港で自動車をホールドまで運転して積み降ろしをする作業者.
- デッキ
  船の内部の階層.
- ホールド
  各デッキ内を一定間隔の領域で仕切られた空間.
- ランプ
  上下デッキに移動するために各デッキの特定ホールドについているスロープ.
- 注文
  乗用車 100 台を港 A から港 B へ輸送, トラック 30 台を港 C から港 D へ輸送というような積載自動車の情報とそれらの積み地と揚げ地に関する情報.
- 積み地
  注文における自動車を積む港.

- 揚げ地
  注文における自動車を降ろす港.

- RT (revenue ton)
  基準となる車一台の面積に対する各注文の車の面積の割合 [2]. 本研究ではこれを資源要求量として扱う.

- ユニット数
  各注文に含まれる自動車の台数.

2.2 入力情報

本研究で扱う二種類の情報について述べる.

- 注文情報
  本稿では注文内に含まれる情報のうち注文番号, 積み地と揚げ地, ユニット数, 積載自動車の RT, 自動車の重量情報を扱う.

- 船体情報
  輸送に使う船のデッキやホールド番号, 各ホールドへアクセスすために通るホールドや各ホールドの許容収容量などの船の内部構造に関する情報.

入力情報のうち注文情報の例を表 2.1 に, 船体情報の例を表 2.2 に示す.

表 2.1: 注文情報の例

注文番号	積み地	揚げ地	ユニット数	RT	重さ (kg)
1	A 港	C 港	20	1.43	1480
2	A 港	C 港	200	1.51	2680
3	A 港	D 港	50	1.57	2580
4	B 港	C 港	14	1.67	1720
5	B 港	D 港	150	1.18	1810

表 2.2: 船体情報の例

デッキ番号	ホールド番号	収容量
1	2	80
1	3	100
2	1	25
2	2	100
2	3	150
3	1	30
3	2	200
3	3	200

2.3 出力

入力情報を元に様々な条件を考慮して注文番号 1 番を 2 デッキ 2 ホールドに 30 台中 30 台割り当てるというような席割を出力する. 出力結果の例を表 2.3 に示す.

表 2.3: 出力結果の例

注文番号	積み地	揚げ地	割当先ホールド	積載台数	ユニット数
1	A 港	C 港	3 階-1	30	30
2	A 港	C 港	2 階-2	40	200
2	A 港	C 港	3 階-2	60	200
2	A 港	C 港	3 階-3	60	200
3	A 港	D 港	2 階-3	50	50
4	B 港	C 港	2 階-1	14	14
5	B 港	C 港	1 階-2	67	150
5	B 港	D 港	1 階-3	83	150

第3章　定式化

この章では本研究における用語や制約，目的関数について説明する．また，各注文に含まれる自動車をどのホールドに何台割り当てるかを最適化する問題を整数計画問題として定式化する．

3.1　記号の定義

本研究における変数と定数記号の定義をする．本研究の変数の定義を表 3.1 に, 定数の定義を表 3.2 に記載する．

表 3.1: 変数の定義

変数	変数の説明
v_{ij}	注文 j をホールド i に n 台割り当てるとき $v_{ij} = n$
x_{ij}	注文 j をホールド i に割り当てるとき $x_{ij} = 1$ そうでないとき $x_{ij} = 0$
n_{it}	港 t でのホールド i の残容量が b'_i のとき $n_{it} = b'_i$
c_{ij}	ホールド i に大きい注文 j を 100 台以下に分割して積むとき $c_{ij} = 1$ そうでないとき $c_{ij} = 0$
$c^1_{i_1 i_2 j}$	ホールドペア (i_1, i_2) に 1. 大きい注文 j_1 を分割して積むとき $c^1_{i_1 i_2 j_1} = p^{\mathrm{c1}}_{\mathrm{large}}$ 2. 小さい注文 j_2 を分割して積むとき $c^1_{i_1 i_2 j_2} = p^{\mathrm{c1}}_{\mathrm{small}}$ 3. それら以外のとき $c^1_{i_1 i_2 j} = 0$
$c^2_{i_1 i_2 j}$	分割した大きい注文 j_1 を 1. 隣のホールドペア (i_1, i_2) に積むとき $c^2_{i_1 i_2 j_1} = n\_p^{\mathrm{c2}}_{\mathrm{large}}$ 2. 同デッキの隣ではないホールドペア (i_1, i_2) に積むとき $c^2_{i_1 i_2 j_2} = s\_p^{\mathrm{c2}}_{\mathrm{large}}$ 分割した小さい注文 j_2 を 3. 隣のホールドペア (i_1, i_2) に積むとき $c^2_{i_1 i_2 j_2} = n\_p^{\mathrm{c2}}_{\mathrm{small}}$ 4. 同デッキの隣ではないホールドペア (i_1, i_2) に積むとき $c^2_{i_1 i_2 j_2} = s\_p^{\mathrm{c2}}_{\mathrm{small}}$ 5. それら以外のとき $c^2_{i_1 i_2 j} = 0$
m_{it}	ホールド i が港 t で作業効率充填率を超えているとき $m_{it} = 1$ そうでないとき $m_{it} = 0$
m_{ijt}	港 t で注文 j をホールド i から積み降ろしを行う際, n 個の通過ホールドで 作業効率充填率を超えそこを v 台車が通るとき $m_{ijt} = vn$

k_{it}^1	ホールド i が港 t で許容充填率を超えているとき $k_{it}^1 = 1$ そうでないとき $k_{it}^1 = 0$
k_{it}^2	ホールド i の残容量が 1RT を超えているとき $k_{it}^2 = 1$ そうではないとき $k_{it}^2 = 0$
k_{it}^3	ホールド i が港 t で許容充填率を超えてそれよりの奥のホールドの中で 1RT を超えたホールドが n 個あるとき $k_{it}^3 = n$ ホールド i が港 t で許容充填率を超えていないとき $k_{it}^3 = 0$
y_{it}^{keep}	ホールド i の中で港 t を通過する注文があるとき $y_{it}^{\text{keep}} = 1$ そうでないとき $y_{it}^{\text{keep}} = 0$
$y_{it_1t_2}$	ホールド i の中で港 t_1 で積んで港 t_2 で降ろす注文があるとき $y_{it_1t_2} = 1$ そうでないとき $y_{it_1t_2} = 0$
$z_{it_1t_2}$	ホールド i の中で港 t_2 を通過する注文があり港 t_2 で降ろす注文の中に港 t_1 で積んだ注文があるとき $z_{i_1i_2t_1t_2} = 1$ そうでないとき $z_{it_1t_2} = 0$
$y_{i_1i_2t}^{\text{load}}$	隣接ホールドペア (i_1, i_2) の中で港 t で積む注文があるとき $y_{i_1i_2t}^{\text{load}} = 1$ そうでないとき $y_{i_1i_2t}^{\text{load}} = 0$
$y_{i_1i_2t}^{\text{keep}}$	隣接ホールドペア (i_1, i_2) の中で港 t を通過する注文があるとき $y_{i_1i_2t}^{\text{keep}} = 1$ そうでないとき $y_{i_1i_2t}^{\text{keep}} = 0$
$y_{i_1i_2t}^{\text{dis}}$	隣接ホールドペア (i_1, i_2) の中で港 t で降ろす注文があるとき $y_{i_1i_2t}^{\text{dis}} = 1$ そうでないとき $y_{i_1i_2t}^{\text{dis}} = 0$
$z_{i_1i_2t}^1$	隣接ホールドペア (i_1, i_2) の中で港 t を通過する注文と積む注文があるとき $z_{i_1i_2t}^1 = 1$ そうでないとき $z_{i_1i_2t}^1 = 0$
$z_{i_1i_2t}^2$	隣接ホールドペア (i_1, i_2) の中で港 t を通過する注文と降ろす注文があるとき $z_{i_1i_2t}^2 = 1$ そうでないとき $z_{i_1i_2t}^2 = 0$

表 3.2: 定数の定義

定数	定数の説明
T	港の集合
T^{c}	チェックポイント港の集合
L	積み地の集合
D	揚げ地の集合
K	船のデッキの集合
J	注文の集合
J^{large}	ユニット数 100 台以上の大きい注文の集合
J^{small}	ユニット数 100 台以下の小さい注文の集合
J_t^{load}	港 t で載せる注文の集合
J_t^{keep}	港 t を通過する注文の集合
J_t^{dis}	港 t で降ろす注文の集合

J_t^{lk}	J_t^{load} と J_t^{keep} の和集合
J_t^{ld}	J_t^{load} と J_t^{dis} の和集合
I	船のホールドの集合
I^{lamp}	ランプがついているホールドの集合
I^{next}	隣同士のホールドペアの集合
I^{same}	同一デッキ内にあるホールドペアの集合
$\overline{I}$	隣接しているホールドペアの集合
I^{num}	船のホールド総数
I_k^{deck}	デッキ k のホールドの集合
I_i^{back}	ホールド i よりも奥のホールドの集合
I_i^*	ホールド i に辿り着くまでに通るホールドの集合
u_j	注文 j に含まれる自動車の台数
g_j	注文 j における自動車一台の重量
a_j	注文 j における自動車一台の RT
b_i	ホールド i の総面積
δ_i^{h}	船の横方向のホールド i の重み
d^{s}	最大許容デッドスペース
f_1^{h}	船の横方向の後方許容値
f_2^{h}	船の横方向の前方許容値
δ_i^{v}	船の縦方向のホールド i の重み
f^{v}	船の縦方向の上方許容値
$\overline{q}_i$	ホールド i における許容充填率
$\overline{q}_i^{\mathrm{s}}$	ホールド i における作業効率充填率
δ_i^{n}	ホールド i の残 RT 利得
p^{c}	変数 c_{ij} のペナルティ重み
p^{k}	変数 k_{it}^3 のペナルティ重み
p^{z}	変数 $z_{it_1t_2}$ のペナルティ重み
p_1^{z}	変数 $z_{i_1i_2t_1t_2}^1$ のペナルティ 1
p_2^{z}	変数 $z_{i_1i_2t_1t_2}^2$ のペナルティ
$p_{\mathrm{large}}^{\mathrm{c1}}$	大きい注文の変数 $c_{i_1i_2j}^1$ のペナルティ
$p_{\mathrm{small}}^{\mathrm{c1}}$	小さい注文の変数 $c_{i_1i_2j}^1$ のペナルティ
$p_{\mathrm{large}}^{\mathrm{sc2}}$	大きい注文の変数 $c_{i_1i_2j}^2$ の緩和量 1
$p_{\mathrm{large}}^{\mathrm{nc2}}$	大きい注文の変数 $c_{i_1i_2j}^2$ の緩和量 2
$p_{\mathrm{small}}^{\mathrm{sc2}}$	小さい注文の変数 $c_{i_1i_2j}^2$ の緩和量 1
$p_{\mathrm{small}}^{\mathrm{nc2}}$	小さい注文の変数 $c_{i_1i_2j}^2$ の緩和量 2
w_1	目的関数 (a) の重み
w_2	目的関数 (b) の重み
w_3	目的関数 (c) の重み
w_4	目的関数 (d) の重み
w_5	目的関数 (e) の重み
w_6	目的関数 (f) の重み

3.2 制約

本研究において，席割を決定する際に考慮するべき独自の制約について記す．

(a) 自動車移動経路に関する制約

ある港において注文が運搬船内の特定のホールドで積み降ろしがあるとき, 自動車がホールドへ到達するためには船の入口とホールドの間で通過する任意のホールドに対して自動車が通るための道を作る必要がある．従って, 本研究では各ホールドに対して自動車の移動通路が十分確保出来る許容充填率を定義し, 任意の港で各ホールドに注文を積むときや各ホールドから注文を降ろすときには, 入口とそのホールドの間で通過する全てのホールドで, その港における充填率が許容充填率以下であることが必要である．本研究ではこれを制約にすることで, 現実的に自動車の積み降ろしが不可能になる席割を禁止する．

(b) 船内重量バランスに関する制約

運搬船で自動車を運ぶ際に船の前方や後方へ自動車を詰め込みすぎたり, 船の上方へ自動車を詰め込みすぎると船が傾いたときにバランスを崩しそのまま横転してしまう可能性がある [7]．本研究ではこの問題を考慮するため, 過去 10 回分の航海データを解析することによりその航海内での運搬船の重量バランスを数値化した．その結果得られた 10 回分の航海データの中で船内重量バランスが最も偏ったときを閾値とし, 運搬船の上下方向と前後方向それぞれに対して, 各港において運搬船から降ろす注文を全て降ろしたとき, または港で積む注文を全て積んだときについて船内重量バランスが閾値を超えてしまうような自動車の割当を禁止する．

3.3 目的関数

席割に関するプランナーやオペレーターとの意見交換や席割作業を実際に体験することで, プランナーは席割を考える上で注文の積み間違えや降ろし間違えが起こらないような席割や, オペレーターにとって効率が良いような席割を目指していることが分かった．本研究では, このような席割を実現するための 6 つの目的関数のパラメータを提案する．

(a) 一つのホールド内の降ろし地を揃える

ある港においてホールドから注文を降ろす場合, 同じホールド内に揚げ地の異なる注文が複数存在していると注文の降ろし間違いが発生する可能性がある．本研究では揚げ地において各ホールド内で揚げ地を通過する注文があるときに, その揚げ地で降ろす注文があるとペナルティが発生する．またこのとき更に降ろす注文の中で積み地が異なる注文が複数含まれていたとする．この場合席割作業の次にシミュレーション作業を行う際, プランナーは降ろし間違えを防ぐため揚げ地が同じ注文は出来るだけホールド内の一つの領域に行き渡るようにホールド内のスペース配分などを考慮しなければならない．このようなプランナーの負担を考え, この場合は降ろす注文の中に含まれる相異なる積み地の分だけ更にペナルティが加算される．

(b) 船内で注文の積み降ろし地を揃える

オペレーターとのヒアリングで, 積み間違いや降ろし間違いを防ぐためになるべく積み地や揚げ地が同じ注文は船内の同じ場所で固めておきたいという要望があった．本研究ではこの要望に応えるために隣接ホールドペアを用意した．隣接ホールドペアとは隣同士に位置するホールド同士のペアと, ランプで繋がったホールド同士のペアのことを指す．各隣接ホールドペアに対して積

み地や揚げ地が異なっている注文が存在していると, その数分ペナルティが発生する. ただし, プランナーの要望で積み地がバラけてしまっても, 揚げ地はなるべく固めたいとの意見をいただいたため, 目的関数 (b) のペナルティ重みは揚げ地のペナルティ重みを積み地よりも重く設定している.

(c) 作業効率充填率を考慮する

ある港で注文の積み降ろしのために自動車をホールドから出すときやホールドに自動車を入れるときに, そのホールドと入口の間で通過するホールドにはその港を通過する注文が一定数存在する. この注文がホールドの領域に対して一定量以上の場合, プランナーはそのホールドを自動車が通れるための道を作る必要がありホールド内における自動車のスペース配分を考える必要がある. またオペレーターは作られた道に自動車を通さなければいけないため, 自動車の移動に慎重になり全体の作業効率が落ちる可能性が高い. 本研究ではこのように作業効率が落ちないための各ホールド毎の作業効率充填率を設定し, これを上回ったホールドを自動車が通過する場合にペナルティが発生する.

(d) デッドスペースを作らない

注文の合計資源要求量が船の合計収容量より十分小さい場合, 特定のホールドに空きスペースが出来る. このスペースが入口付近にあれば追い積みという港での突然の追加注文を積むことに対応出来るが, 空きスペースが出来たホールドに到達するまでに通るホールドが許容充填率を上回っている場合は, このスペースに追い積みをすることは出来ない. このような空きスペースのことをデッドスペースと言う. 目的関数 (d) ではランプで繋がった船のホールドが許容充填率を超え特定デッキ間の移動ができない状態になったとき, そのホールドよりも奥にあるホールドの中で, 指定された最大許容デッドスペースよりも大きいデッドスペースがあるホールドが存在するときに大きいペナルティが発生する.

(e) 残容量をなるべく入口付近に寄せる

船の入口に近いホールドにまだ自動車を入れることのできるスペースがあると船への追い積みに対応できる. また, 研究で扱う船は追い積みに対応しやすい 5 デッキに残容量をなるべく残したいとのプランナーから要望があった. 目的関数 (e) では, 積載自動車を全て積み終わる最後の積み地をチェックポイントとし, チェックポイントで 5 デッキや入口に近いホールドに対して残容量が多いほど全体目的関数のペナルティを減らす利得が発生する.

3.4 定式化

$$
\begin{aligned}
\min \quad & w_1 p^{\mathrm{z}} \sum_{i\in I}\sum_{t_1\in L}\sum_{t_2\in D} z_{it_1t_2} + w_2 \sum_{i_1\in I}\sum_{i_2\in I}(p_1^{\mathrm{z}} \sum_{t_1\in L} z^1_{i_1i_2t_1} + p_2^{\mathrm{z}} \sum_{t_2\in D} z^2_{i_1i_2t_2}) \\
& + w_3 \sum_{i\in I}\sum_{j\in J}\sum_{t\in T} m_{ijt} + w_4 p^{\mathrm{k}} \sum_{i\in I^{\mathrm{lamp}}}\sum_{t\in L} k^3_{it} - w_5 \sum_{i\in I}\sum_{t\in T^{\mathrm{c}}} \delta^{\mathrm{n}}_i n_{it} && (3.1)
\end{aligned}
$$

$$
\begin{aligned}
\text{s.t.} \quad & x_{ij} \in \{0,1\}, y^{\mathrm{keep}}_{it} \in \{0,1\}, y_{it_1t_2} \in \{0,1\}, z_{it_1t_2} \in \{0,1\}, y^{\mathrm{load}}_{i_1i_2t} \in \{0,1\}, \\
& y^{\mathrm{keep}}_{i_1i_2t} \in \{0,1\}, y^{\mathrm{dis}}_{i_1i_2t} \in \{0,1\}, m_{it} \in \{0,1\}, k^1_{it} \in \{0,1\}, k^2_{it} \in \{0,1\}, \\
& 0 \le z^1_{i_1i_2t}, 0 \le z^2_{i_1i_2t}, 0 \le m_{ijt}, 0 \le k^3_{it}, 0 \le n_{it}, 0 \le v_{ij} \le u_j, \\
& \qquad\qquad \forall i \in I, \forall j \in J, \forall t \in T. && (3.2)
\end{aligned}
$$

$$\sum_{i\in I} x_{ij} = 1, \qquad j \in J^{\mathrm{small}}. \quad (3.3)$$

$$\sum_{i\in I} v_{ij} = u_j, \qquad j \in J. \quad (3.4)$$

$$\sum_{j\in J} a_j v_{ij} \le b_i, \qquad i \in I. \quad (3.5)$$

$$\sum_{j\in J_t{}^{\mathrm{keep}}} \frac{a_j v_{ij}}{b_i} \ge \overline{q}^{\mathrm{s}}_i + m_{it}, \qquad i \in I, t \in T. \quad (3.6)$$

$$\sum_{i_2\in I^*_{i_1}} m_{i_2t} v_{i_1j} \le m_{i_1jt}, \qquad i_1 \in I, j \in J^{\mathrm{ld}}_t, t \in T. \quad (3.7)$$

$$\sum_{j\in J^{\mathrm{lk}}_t} \frac{a_j v_{ij}}{b_i} \le k^1_{it} + \overline{q}_i, \qquad i \in I^{\mathrm{lamp}}, t \in L. \quad (3.8)$$

$$\sum_{j\in J^{\mathrm{lk}}_t} \frac{a_j v_{ij} + d^{\mathrm{s}}}{b_i} \ge 1 - k^2_{it} \qquad i \in I, t \in L. \quad (3.9)$$

$$\sum_{j_1\in J_t{}^{\mathrm{keep}}} \frac{a_{j_1} v_{i_1j_1}}{b_{i_1}} \le \overline{q}_{i_2} + 1 - x_{i_2j_2}, \qquad i_1 \in I^*_{i_2}, i_2 \in I, j_2 \in J^{\mathrm{ld}}_t, t \in T. \quad (3.10)$$

$$\sum_{j\in J^{\mathrm{lk}}_t} a_j v_{ij} \le b_i - n_{it}, \qquad i \in I, t \in T^{\mathrm{c}}. \quad (3.11)$$

$$\sum_{i_2\in I^{\mathrm{back}}_{i_1}} k^2_{i_2t} + I^{\mathrm{num}}(k^1_{i_1t} - 1) \le k^3_{i_1t}, \qquad i \in I^{\mathrm{lamp}}, t \in L. \quad (3.12)$$

$$\sum_{i_1\in I^{\mathrm{deck}}_{k_1}} x_{i_1j} \sum_{i_2\in I^{\mathrm{deck}}_{k_2}} x_{i_2j} \le 0, \qquad j \in J^{\mathrm{large}}, k_1, k_2 \in K, k_1 \ne k_2. \quad (3.13)$$

$$\frac{v_{ij}}{u_j} \le x_{ij}, \qquad i \in I, j \in J. \quad (3.14)$$

$$f^{\mathrm{h}}_1 \le \delta^{\mathrm{h}}_i g_j v_{ij} \le f^{\mathrm{h}}_2, \qquad i \in I, j \in J^{\mathrm{keep}}_t \cup J^{\mathrm{lk}}_t, t \in T. \quad (3.15)$$

$$\delta^{\mathrm{v}}_i g_j v_{ij} \le f^{\mathrm{v}}, \qquad i \in I, j \in J^{\mathrm{keep}}_t \cup J^{\mathrm{lk}}_t, t \in T. \quad (3.16)$$

$$x_{ij} \le y^{\mathrm{keep}}_{it}, \qquad i \in I, j \in J^{\mathrm{keep}}_t, t \in T. \quad (3.17)$$

$$x_{ij} \le y_{it_1t_2}, \qquad i \in I, j \in J^{\mathrm{load}}_{t_1} \cap J^{\mathrm{dis}}_{t_2}, t_1, t_2 \in T. \quad (3.18)$$

$$z_{it_1t_2} \le y^{\mathrm{keep}}_{it_2}, \qquad i \in I, j \in J^{\mathrm{keep}}_{t_2}, t_1, t_2 \in T. \quad (3.19)$$

$$z_{it_1t_2} \leq y_{it_1t_2}, \qquad i \in I, j \in J^{\text{keep}}_{t_2}, t_1, t_2 \in T. \quad (3.20)$$

$$y_{it_1t_2} + y^{\text{keep}}_{it_2} - 1 \leq z_{it_1t_2}, \qquad i \in I, t_1, t_2 \in T. \quad (3.21)$$

$$x_{i_1j} + x_{i_2j} \leq 2y^{\text{load}}_{i_1i_2t}, \qquad i_1, i_2 \in \overline{I}, j \in J^{\text{load}}_t, t \in T. \quad (3.22)$$

$$x_{i_1j} + x_{i_2j} \leq 2y^{\text{keep}}_{i_1i_2t}, \qquad i_1, i_2 \in \overline{I}, j \in J^{\text{keep}}_t, t \in T. \quad (3.23)$$

$$x_{i_1j} + x_{i_2j} \leq 2y^{\text{dis}}_{i_1i_2t}, \qquad i_1, i_2 \in \overline{I}, j \in J^{\text{dis}}_t, t \in T. \quad (3.24)$$

$$y^{\text{load}}_{i_1i_2t} + y^{\text{keep}}_{i_1i_2t} - 1 \leq z^1_{i_1i_2t}, \qquad i_1, i_2 \in \overline{I}, t \in L. \quad (3.25)$$

$$y^{\text{dis}}_{i_1i_2t} + y^{\text{keep}}_{i_1i_2t} - 1 \leq z^2_{i_1i_2t}, \qquad i_1, i_2 \in \overline{I}, t \in D. \quad (3.26)$$

第4章　整数計画問題に対する解法

鵜川ら [6] は，第 3 章で定式化した整数計画問題に対して，整数計画ソルバーを用いて実行可能解を得ることを提案した．

計算実験の結果，規模の小さい問題例に対しては有効な解を得ることを確認したが，注文数や積み地や揚げ地の数の増加とともに制約や組合せが急速に増加してしまい，指定された時間内では十分に計算が進まない可能性が指摘されている．

4.1　求解が可能な問題例の規模

注文数や港の数を変更した問題例をいくつか準備し，どの程度の規模の問題例ならばある程度の時間内で解けるのか調査を行った．実験に用いるプログラムは Python を用いて実装し, 計算機は PowerEdge T320 (CPU: Intel(R) Xeon(R) CPU E5-1410 v2 (2.80 GHz, 10 M cache), RAM: 96 GB) を使用した．また，整数計画ソルバーとして Gurobi Optimizer (ver 9.0.0) を使用した．

表 4.1 に実行可能解が初めて出力された計算時間と，計算を回して 1 時間後と 24 時間後の解の精度を示した．

結果より，注文数が 250 程度，港の数が 3,4 より少なければ，24 時間以内にある程度の解を出力することが確認できた．一方で，注文数が 350 の問題例に関しては，24 時間後でも解の探索があまり進まなかった．

実際にはこれ以上の注文数を持つ問題例が多く存在しており，整数計画ソルバーを用いる手法では，より大規模な問題例を解くことは難しいと考えられる．

表 4.1: 複数の規模の問題例に対する計算結果

問題例			実行可能解が出力された時間	1 時間後の解	1 時間後の下界値	24 時間後の解	24 時間後の下界値
注文数	積み地の数	揚げ地の数					
109	2	3	537	−4301.36	−4461.98	−4388.44	−4461.98
109	2	5	1674	−3522.82	−4461.98	−4264.69	−4461.98
109	4	3	344	−3515.46	−4461.98	−4365.89	−4461.98
250	2	3	2381	−3155.02	−4461.98	−4083.18	−4461.98
350	2	3	2249	−2738.00	−4461.98	−2883.58	−4461.98

第5章　ヒューリスティックを用いた解法

4.1 章で，現実的な時間で実行可能解を出力することのできる問題例の規模を確認した．

より大規模な問題例を現実的な時間で解くために，ヒューリスティックを用いた数理モデルを提案する．

5.1　ホールドの統合

本研究では，ホールドの数が 43 の運搬船を考えている．実オペレーションでは，それぞれのホールドを個別で考えるのではなく，複数のホールドをある程度まとめて考えて作業を行っている．そのため，各階においてスロープのついているホールドを境目として，2 つのホールドのまとまりとして考える．

複数のホールドを 1 つのまとまりとして考えると，43 のホールドを，17 のまとまりに統合することができる．このまとまりをセグメントと呼び，具体的な統合については図 5.1 に示す．

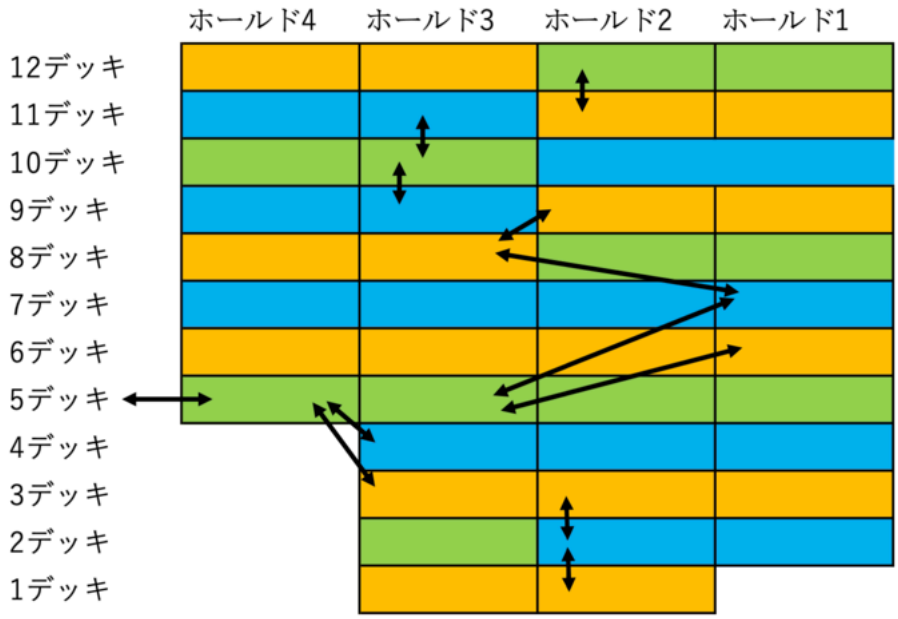

図 5.1: 統合後のホールドを色付けしたもの

5.2 解の表現方法

鵜川らによる研究の数理モデル [6] では，ホールドに注文を何台割り当てる，という連続的な解の表現方法を用いていた．ヒューリスティックモデルでは，このような連続的な解表現方法を用いず，注文のセグメントへの割当のみを最適化し，詳細な車両の積載は決め打ちのルールを基に行っていく．積載のルールは，ホールドの積載可能量, 許容充填率などを満たすように設計を行っていく．

このように積載貨物を決めていくことで，注文を複数のホールドに渡って分割する場合であっても，隣接したホールドに車両を積むことができるため，プランナーの考える席割に近い解を出力することができる．

5.3 局所探索法

局所探索法とは, ある解に少しの変更を加え得た近傍解が元の解よりも良い解である場合に, 変更後の解に移動するという操作を良い解が見つからなくなるまで繰り返す方法である. 変形を加える操作を近傍操作と呼び, 近傍内に改善解を持たない解を局所最適解と呼ぶ.

本研究では，初期解を生成した後に局所探索法を用いて解の改善を行う手法を提案する．また，評価関数として鵜川ら [6] によって提案された目的関数を用いる．また，貨物の重量バランスに関する制約などに違反している場合には，制約違反度合いをペナルティ関数として表し, 目的関数に加えて探索を行う．

5.4 初期解の生成

局所探索における初期解の生成方法として，2 種類のアプローチを提案する．

5.4.1 整数計画問題の緩和解の利用

定式化を行った整数計画問題の整数制約を緩和した線形緩和問題の解を初期解として利用することを考える．この手法は，有効な解が得られたことが確認できている整数計画問題の解と線形緩和問題の性質が似ている場合に有効であると考えられる．予備実験としてハミング距離を用いてそれぞれの解の性質を比較する．

ハミング距離

ハミング距離とは，2 つのベクトルの各要素の比較を行い，異なる要素の数がどの程度あるかを示すものである．

長さ n の 0-1 ベクトル x, y に対して，ハミング距離 d_H は一般的に以下のように定義される．

$$d_H(x, y) = \sum_{i=1}^{n} |x_j - y_i|$$

ハミング距離の計算方法

本研究では，各注文におけるホールドへの割当台数を目的変数としている．各注文における車両台数は異なるため，ハミング距離を計算するためには正規化をする必要がある．

注文における車両台数のなかで，特定のホールドに割り当てる注文台数の割合を変数とすると，0 から 1 の値を必ずとることがわかる．各注文において，注文が全て割り当てられていれば和は 1 になる．

整数計画問題の解と線形緩和問題の解をそれぞれ正規化したベクトルを x, y とする．注文数を n，ホールドの数を m としたとき，ハミング距離を以下のように計算する．

$$d_H(x, y) = \sum_{i=1}^{n} \sum_{j=1}^{m} |x_{ij} - y_{ij}|$$

注文の全てが割り当てられている場合にハミング距離の和は $2n$ になるため，$2n$ で割ると 2 つの解の性質がどの程度異なるかを測ることができる．

線形緩和問題の解の性質

整数計画問題の解と線形緩和問題の性質の比較を行う．解の精度が十分に良いことがわかっている解で比較を行う必要があるため，整数計画問題において厳密解が出た問題例で比較を行い，結果を表 5.1 に示す．

どのインスタンスにおいても，整数計画問題の解の性質に近い線形緩和問題の解が存在することが確認できた．

表 5.1: 線形緩和問題の解との比較

注文数	積み地	揚げ地	解の差異 (%)
38	2	3	0.19
63	2	3	3.57
57	2	5	1.07
66	4	3	0.71

線形緩和解の丸め方

線形緩和問題では，注文のホールドへの割当として小数点が出てくる場合がある．各注文を，緩和解の変数値の最も大きいホールドに割り当てると一部のホールドに大量に割り当てられる解が出る可能性があるため，割り当てられた値を確率として，その確率を基にランダムに割り当てる手法で初期解を生成する．ただし，割り当てるホールドの容量制約が満たされない場合には，そのホールドには割り当てない．

5.4.2 港が同じ注文をまとめて席割を作成

鵜川らは，注文の積み地と揚げ地が同じ注文を 1 つのグループとして考え，グループ化された注文をホールドに割り当てることで計算時間を短縮する手法を提案した [6]．解の精度としては，ある程度有効ではあるものの，実際に利用するためには修正が必要であった．

本研究では，この手法で得られた解を局所探索における初期解として利用するアプローチを提案する．

5.5 近傍操作

近傍操作には挿入近傍操作と交換近傍操作を用いる．

挿入近傍操作は，セグメントに割り当てられている注文の 1 つを，ランダムに異なるセグメントに挿入する操作である．交換近傍操作は，異なるセグメントに割り当てられている注文を 2 つランダムに選び，それらを入れ替える操作である．

本研究では，はじめに挿入近傍操作を改善がなくなるまで繰り返し，その後交換近傍操作を改善がなくなるまで行う．挿入近傍操作で 1 回でも改善があった場合は，再び挿入近傍操作からやり直す．

5.5.1 計算時間の短縮

本研究では，局所探索における近傍操作の 1 つとして挿入近傍操作を用いるが，操作に工夫を加えることで解の精度を変化させることなく計算時間を短縮させることができると考えられる．

挿入近傍操作では，1 つの注文を異なるセグメントに挿入する．その際に，挿入可能な位置すべてを探索し評価関数の値が最も良くなる位置に挿入する．

しかし，あるセグメントに格納される注文の順番が変化しても評価関数の値が全く変化しない場合がある．そのような場合にはすべての挿入可能位置を探索する必要がないため，計算時間を短縮することができる．

以下に計算時間を短縮する提案手法の概要を示す．

Algorithm 1 計算時間を短縮する手法

```
1: あるセグメントから，注文を 1 つ選択する
2: 挿入先のセグメントの，一番最後の位置に挿入する
3: 複数のセグメントに分割された注文を J_split とする．
4: for j ∈ J_split do
5:     j の前後に注文を挿入する
6:     if 評価関数の値が暫定解よりも良い then
7:         暫定解を更新
8:     end if
9: end for
10: if 暫定解が探索における最良解よりも良い then
11:     最良解を更新
12: end if
```

第6章　計算実験

6.1　実験環境

実験に用いるプログラムは Python を用いて実装し, 計算機は PowerEdge T320 (CPU: Intel(R) Xeon(R) CPU E5-1410 v2 (2.80 GHz, 10 M cache), RAM: 96 GB) を使用した．また，整数計画ソルバーとして Gurobi Optimizer (ver 9.0.0) を使用した.

6.2　計算時間の短縮

5.5.1 で解の精度を変化させることなく計算時間を短縮する手法を提案した．計算実験を行い，挿入可能な位置にすべて挿入した場合と提案手法において，計算時間の秒数の比較を行う．

挿入近傍操作を一定回数行った際の計算時間の比較結果を表 6.1 に示す.

提案手法により、全てのインスタンスにおいて計算時間が短縮できることを確認した．解の精度を落とすことなく計算時間を短縮することができるため，有効な手法だと考えられる.

6.3　初期解による比較

5.4 章で提案した初期解生成の手法と，注文をランダムに割り当てた初期解の比較を行う．5.4.1 章の緩和解を用いる方法を手法 1，5.4.2 章の港ごとに注文をまとめる方法を手法 2, 初期解をランダムに生成する方法を手法 3 と記し, 計算結果を表 6.2 に示す.

手法 1 では，他の手法に比べて多くの計算時間を要していることを確認した．これにより，最適化変数の整数制約を緩和した問題であっても計算時間はあまり短くならないことが確認できた.

ランダムに初期解を生成した手法 3 では，手法 1 と比較して短い計算時間である程度の精度の解を得る事ができることを確認した.

6.4　2 つの提案手法の比較

5 章で提案したヒューリスティックを用いる手法と，整数計画問題として解いた結果の比較を行う．整数計画問題を解く手法では，探索を行って 1 時間後と 24 時間後の評価関数の値を記す．ヒューリスティックを用いる手法では，6.3 章で確認した最も解の精度が良い手法を記す．計算結果を表 6.3 に示す.

結果より，計算時間を短縮しつつ，ある程度の精度の解を得ることができていることが確認できた.

表 6.1: 近傍操作に関する計算時間の比較

注文数	積み地	揚げ地	全ての位置を探索	提案手法
109	2	3	145	81
109	2	5	192	89
109	4	3	122	99
250	2	3	387	204

表 6.2: 初期解生成による計算時間と解の比較

問題例			手法 1		手法 2		手法 3	
注文数	積み地	揚げ地	計算時間	解	計算時間	解	計算時間	解
109	2	3	1641	-2179	-	-	873	-2975
109	2	5	8057	-594	-	-	938	-1167
109	4	3	8782	1840	-	-	728	-1378
250	2	3	9726	-2755	-	-	3069	-2235

表 6.3: 提案手法による比較

問題例			整数計画問題の解		ヒューリスティック	
注文数	積み地	揚げ地	1 時間後の解	24 時間後の解	計算時間	解
109	2	3	-508	-3478	873	-2975
109	2	5	-2612	-3354	938	-1167
109	4	3	1240	-3455	728	-1378
250	2	3	実行不可能	-2064	3069	-2235

第7章　まとめ

自動車運搬船における自動車を船に割り当てる席割の作成に関して，プランナーやオペレータとのヒアリングを基に定式化を行い 2 つの数理モデルを提案した．

整数計画ソルバーを用いて問題を解く際に，有効な解を出力することのできる問題例の規模を確認した．また，注文数が増えた場合や更なる制約や目的関数を追加する場合に計算時間が膨大に増大しないことを目指し，ヒューリスティックを用いた新たな数理モデルを提案した．

新たに提案した数理モデルの局所探索における近傍操作に関して，挿入近傍操作において解の精度を落とすことなく計算時間を短縮する手法を提案し，有効性を確認した．また，局所探索における初期解を生成する手法を 2 種類提案し，計算実験を行い比較を行った..

ヒューリスティックを基にした新たな数理モデルを用いることで，整数計画問題を解く手法と比較して，計算時間の短縮及びある程度の精度の解を得ることができることを確認した．

謝辞

本研究の遂行にあたり, 熱心な指導と助言を頂きました柳浦睦憲教授に深く感謝の意を表します. 提案手法の検討やその有用性において活発に議論を頂きました小野廣隆教授，大舘陽太准教授に大変お世話になりました. 深くお礼申し上げます. 日々の研究室生活においては柳浦研究室の皆様にお世話になりました. 皆様のおかげで有意義な研究活動に勤しむことができました. 深くお礼申し上げます.

参考文献

[1] 「商船三井 プレスリリース」https://www.mol.co.jp/pr/2019/19072.html (2021 年 12 月 3 日)

[2] 「レベニュートン (Revenue Ton)」http://www.b-plaza.jp/news/detail.php?p=12&n=5971&gid=t11roro40ft (2021 年 12 月 3 日)

[3] 「日本橋海運 貿易関連用語集」http://nihonbashi-shipping.co.jp/lexicon/50_su/

[4] 株式会社構造計画研究所, 貨物席割システム, 特願 2001–371362,2001-12-5

[5] 斉藤努，柳田俊樹, 自動車船積付支援システムの自動席割について, オペレーションズ・リサーチ学会，48 (2003) 216–217.

[6] 鵜川知哉, 自動車運搬船における貨物積載プランニングの席割問題に対する数理モデリング, 修士論文，名古屋大学情報学研究科, 2020

[7] Eivind Wathne, Cargo Stowage Planning in RoRo Shipping, Master thesis, Norwegian University of Science and Technology, Marine Technology, 2012.