造船所内ブロックストックヤードの スケジューリングにおけるブロック配置の最適化

正会員　　木村　元*

Optimization of arrangement of scheduled blocks in a ship-building stockyard

by　　Hajime Kimura, *Member*

Key Words:　*Block Stockyard, Block Placement System, scheduling optimization*

1. 緒　　言

造船所では，建造量を増やすためにドッグにおける建造期間の短縮に努めており，そのためにはブロック総組およびドックでのブロック搭載を迅速に行うことが重要である．一方で，生産の効率化の見地からは工場の設備や人員が一定であることが理想的であるため，ブロックの建造能力もほぼ一定となることから，総組およびブロック搭載日にジャスト・イン・タイムにブロックを建造・供給することは不可能である．そこで，大組ブロックをストックヤードにストックしておく必要が生じる．しかし、船舶の建造ブロックは巨大であるため一般的な機械部品のように倉庫のラックにストックしたり，コンテナのように積み重ねることが不可能である上，品数も多いために広大なストックヤード用の敷地が必要となる。残念ながら国土が狭く地価が高い我が国の造船所では，十分な広さのストックヤードを確保することは困難である．ブロックを台車で運搬する場合、ストックヤードの形態としては，必要なブロックをすぐに取り出せるようブロック置場の全てが運搬用の通路に面していることが理想的だが，ストックヤード面積を節約し，より多くのブロックを置けるようにするために，通路に面した置場の奥の方に，通路に面さない別の置場を設定し，手前の置場を通ってブロックを出し入れする，スタックと呼ばれる先入れ後出し(First In Last Out: FILO)の構造にすることが多い．しかし，スタックによる蔵置では，全ブロックの出入庫スケジュールに合わせて巧みに配置場所を決めてやらない限り，奥のブロックを取り出す際に手前のブロックをどかすという無駄がほぼ必然的に発生するという問題がある．造船所の規模や扱うブロック数によっては，ブロック蔵置場所の決定を現場任せにしておくと，ストックヤード内を四六時中台車が右往左往するような事態を招いてしまう．本研究の対象としている造船所では，大部分のスタックの深さが 2～4 段であり，また最大で 6 段ものスタックが存在している．そのためスタック構造まで考慮に入れ，無駄の生じないブロックの蔵置場所を自動的に計画するアプリケーションソフトが求められる．しかし，ブロックを置く場所を視覚的に検討・管理するためのツール [1)]は存在したが, 自動的にブロック配置を計画するシステムは存在しなかった．

そこで本研究では，スタック構造を持つストックヤードへのブロック蔵置問題を組み合わせ最適問題として帰着し，分枝限定法を用いて解決を図る．また，ブロック形状ごとに置場を区別し，工程にエラーが発生したときに再スケジュールできるような実用的なシステムの構築を目指す．実際の造船所のデータを元にシミュレーションを行い，提案手法の有用性を確認する．

2. 問　題　設　定

2.1 スタック構造とは

スタック構造とは，情報工学におけるデータの保持方法としてデータを後に入れたものを先に出すような構造を表す．ここではデータではなく物理的なブロックを扱うが，シミュレータ上で考えれば全く同一である．

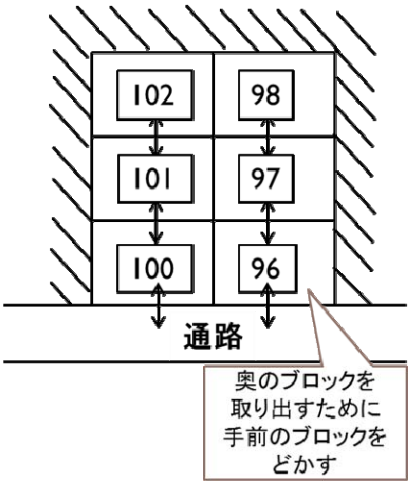

Fig. 1 A stack structure in a stockyard.

例えば Fig.1 のようにブロック置場にブロックがストックされているとする．このとき，102 番のブロックを取り出すためには，それよりも通路側に置かれている 100 番と 101 番のブロックを一旦通路へ出して別の置場へどかさなければならない．このとき，ブロックを支えている脚あるいは架台および台車の都合により，たとえ隣の 96, 97, 98 番のブロックが置かれていないとしても 101 番のブロックを右側にずらしてから通路へ運び出すことはできない．よって Fig.1 の 100, 101, 102 番ブロックは, 102, 101, 100 番の順にスタックへ入り，出るときは逆に 100, 101,102 番の順でなければならない．これが先入れ後出し（FILO）のスタック構造である．

蔵置される全ブロックは互いに区別され，それぞれ蔵置開始日と蔵置終了日が予め決められている．蔵置期間が最短の場合，蔵置開始日の翌日が蔵置終了日となる．

* 九州大学　大学院工学研究院

原稿受付　(学会にて記入します)
春季講演会において講演　(学会にて記入します)

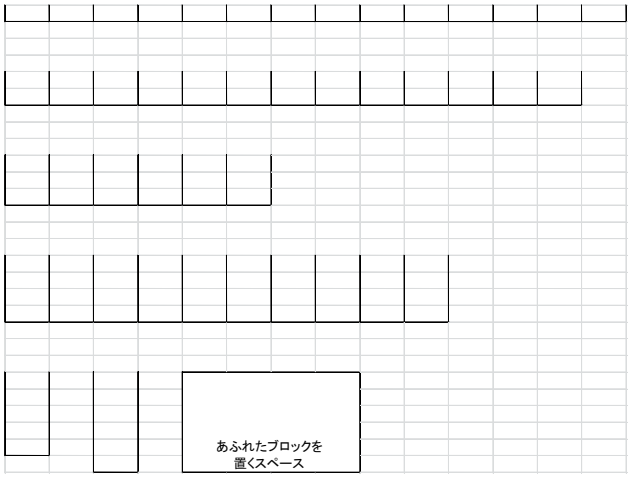

Fig. 2 A model of a real stockyard.

本研究の対象としている造船所では，Fig.2 に示すように深さ 1 段のスタック（普通スタックとは呼ばないが）が 14 組，深さ 2 段のスタックが 13 組，深さ 3 段のスタックが 6 組，深さ 4 段のスタックが 10 組，深さ 5 段と 6 段の深さのスタックがそれぞれ 1 つずつ，合計で 45 組のスタック，109 個分のブロック置場となっている．ここで解くべき問題とは，蔵置期間が決まっている全ブロックをその期間中にどのスタックへ置くかを割り当てることである．ストックヤード内でのブロックの無駄な移動を避けるためには，ブロックをその蔵置期間中ずっと同一スタックで蔵置しておくことが理想的である．

2. 2 無駄なくスタックへ蔵置するためのルール

スタックではブロックの出し入れは FILO でなければならないが，これをスタックへ蔵置されるブロック同士の蔵置期間についての制約条件として定式化する．任意の 1 つのスタックについて，ブロックの蔵置がスケジュールされていると仮定し，ある時刻 t（本研究の場合 1 日単位）においてスタックの深さ d（1 が最も深く，数字が増加するほど浅い置場とする）に置かれているブロックの蔵置期間に注目する．その蔵置開始時刻を bs(t,d)と表し，蔵置終了時刻を be(t,d)と表す．ここで，時刻 t と深さ d で指定された深さと時刻に何も置かれていない場合は bs(t,d)=be(t,d) = t とする．このとき，ブロックの出し入れが FILO となるための条件は、全期間における任意の t において bs(t,d1) ≦ bs(t,d2) かつ be(t,d1) ≧ be(t,d2) ただし深さ d1≦d2 を満たすことである．本研究ではこの条件を「理想スタック蔵置ルール（ideal stacking rule）」と呼ぶ．

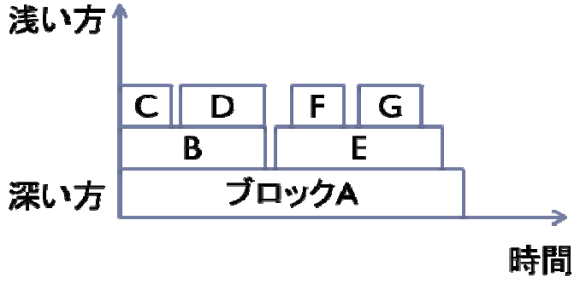

Fig.3 An example of an effective schedule following the ‘ideal stacking rule’ in a stack.

Fig.3 は上記の理想スタック蔵置ルールを満たすような蔵置期間のブロックを選んで同じスタックへ置いた例を示す．図の縦軸方向はスタックの深さ d を示すが，視覚的に理解するために上方向ほど d が大きく浅いことを示す．横軸はスケジュールの時刻を表し，図中のＡ～Ｇの矩形は各ブロックの蔵置期間を表す．つまり，各ブロックを蔵置期間中にスタックのどの深さに置くのかを示す．この Fig.3 のような置き方をすれば，奥に置かれたブロックを取り出すために手前のブロックを動かす必要は無い．

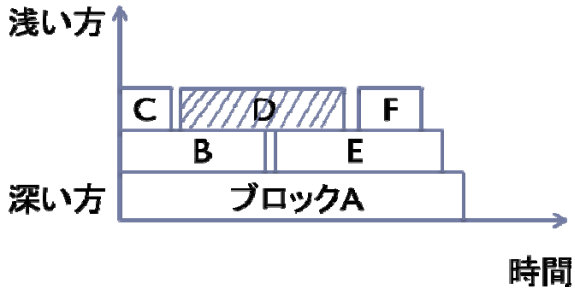

Fig.4 An example of a bad schedule which does not follow the ‘ideal stacking rule’ in the stack.

一方，Fig.4 は理想スタック蔵置ルールを満たさない置き方の例を示している．Fig.4 のブロック B をスタックから取り出す場合にはブロック D を一時的にどかす必要が生じてしまう．よってこのスタックに蔵置期間がブロック D のようにブロック B と E の期間をまたぐものをこれらより浅い位置へ置くことは不適当である．

3. ブロック割当アルゴリズムの提案

本章では，蔵置期間が決まっているブロックを蔵置するスタックを決定するアルゴリズムを提案する．

3. 1 前準備：蔵置予定ブロックデータのソート

2.2 節で説明した理想スタック蔵置ルールを満たすようにスタックへブロックを置いた場合，同一スタックに出入りするブロックの蔵置期間は必ず以下を満たす：

1）蔵置開始日が異なるブロック同士を比較した場合，蔵置開始日が早い方のブロックが必ず先に入る．

2）蔵置開始日が同じブロック同士を比較した場合，蔵置期間が長い方のブロックが必ず先に入る．

以上より，置き場所を決めるブロックの検討順序として，まず蔵置開始日の昇順にソートし，さらに蔵置開始日が同じブロック間においては蔵置期間の降順にソートすることにより，理想スタック蔵置ルールに従う置き方の探索範囲を大幅に狭め，スケジュール効率を上げることができる．Fig.5 は 43 個分のブロックの蔵置期間について上記のソート方法を用いて実データをソートした例を示す．Fig.5 の縦軸は各ブロックの区別を表し，横軸は時間を表す．水色の矩形は，各ブロックの蔵置期間を表している．このような蔵置期間順にソートした後で，Fig.5 の上のブロックから順に置くべきスタックの割当を検討していけば良い．

3. 2 ソートされたブロックの割当アルゴリズム

(1) 深いスタックから優先して埋める分枝限定法 1

理想スタック蔵置ルールを遵守したブロックの置き方

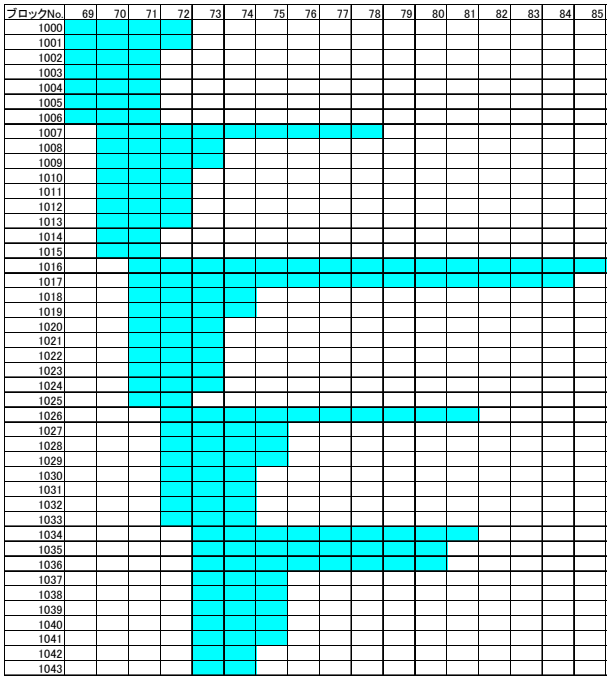

Fig. 5 An example of the sorted stock periods of blocks.

を考える場合，深いスタックほど扱いが厄介であることは直感的にも自明である．そこで，3.1 節でソートされたブロック順にまず最も深いスタックを空きスペース無くきっちり埋めるようブロックを選択していき，スタックへ置くことが決まったブロックは先のソートされたブロック順列から削除していく．こうして最も深いスタックが埋まったら，次に深いスタックを選択し，再び残ったソート順列ブロック順にきっちり埋まるようブロックを選択していくという操作を繰り返す．すると，浅いスタックにあまりブロックが置かれずスカスカになったり，理想スタック蔵置ルールを満たせず置場の決まらないブロックが残るが，深さが浅いスタックは扱いやすく，また残ったブロックは後述のように蔵置期間を分割して空いたスタックへ蔵置するので修正は容易である．

ここで，配置対象のスタックに対し，3.1 節でソートされたブロック順列中のどのブロックを選択するのかについては，可能な組み合わせを全数探索したのでは天文学的な時間がかかってしまうため，分枝限定法を用いる．これは，可能な組み合わせのうち，スタックの段数などから同時に蔵置が不可能なブロックを即座に検出できることなどを利用している．また，探索の途中で明らかに無駄と思われる割当案を排除するため，探索途中での配置案に生じた延べ空き置場数を解候補の評価値とし，探索の枝刈りに利用する方法を考案した．

Fig.6 および Fig.7 に上記の探索途中の空きスペースを評価した例を示す．ここで，図の縦軸は注目しているスタックの深さ（ただし上ほど浅い）を表し，横軸は時刻を表す．1～6 の番号が振られた矩形は，各ブロックの蔵置期間を表す．ここで，分枝限定法アルゴリズムは現在 6 番目に蔵置するブロックについて，Fig.6 の候補と Fig.7 の候補を検討している状況であるものとする．このとき 6 番目のグレーのブロックを選んで蔵置した場合，これ以降，図中のハッチングされた空き領域を埋めることはできない．そこでこの空き領域をコストとして評価する．

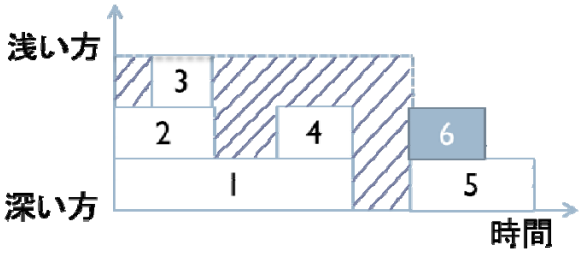

Fig. 6 An example of the cost function used for the cutting branches in the search tree.

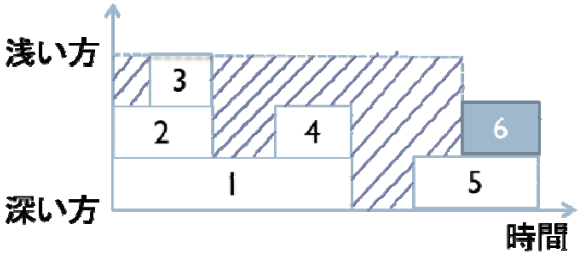

Fig. 7 An example of the cost function used for the cutting branches in the search tree.

Fig.6 と Fig.7 のハッチングされた空き領域を比較すると，Fig.6 のほうが優れることになるが，あくまでも探索途中段階での評価なので，これが小さい Fig.6 のほうが全期間でのスケジュールにおいて最適になることは保障されない点に注意を要する．

(2) ソートされたブロック順に置くべきスタックを強制的に割り当てる分枝限定法２

この方法では，スタックの深さは関係なく 3.1 節でソートされたブロック順に，理想スタック蔵置ルールを遵守して置けるスタックがある限り強制的にそこへ置いていく方法である．理想スタック蔵置ルールを遵守して置けるスタックは複数存在する場合があるので，これを分枝限定法を用いて探索する．また，どのスタックに置いても理想スタック蔵置ルールを遵守できない，あるいは物理的に置くことが不可能な場合はそのブロックの配置をあきらめ，ソート順列中の次のブロックを割り当てる．分枝限定法で解候補を探索する際，やはり膨大な組み合わせの探索を抑制するために(1)で用いた空きスペースによる評価を利用して適度に枝刈りを行う．

3. 3 理想スタック蔵置ルールを遵守した場合に配置できなかったブロックの分割蔵置

ブロック数に対して置場やスタック数の余裕がないと，理想スタック蔵置ルールに従った蔵置ができないブロックが発生する．これは好ましいことではないが，このようなブロックはその蔵置期間中にストックヤード内を移動することになる．Fig.8 はあるスタックにおいて理想スタック蔵置ルールを遵守するようブロックをスケジュール後，スタックの最も浅い置場が空いている期間に，配置できなかった別のブロックを蔵置させた場合の例である．Fig.8 の赤い矩形で示すとおり，蔵置期間を分割してストックヤード内での移動が 1 回発生するたびに必要な置場が 1 つ増加してしまうことが分かる．これは不可避ではあるが，このような必要な置場の増加を抑制するた

め，蔵置できていないブロックの蔵置期間の分割パターンを最適化する必要がある．残念ながら本研究ではそこまでのシステムを構築できていない．今のところ暫定的に，各スタックの空き期間をサーチして，置き切れなったブロックの蔵置期間と上記の空き期間とが重なる期間が最大になるブロックを選択して蔵置期間を分割・蔵置していくようになっている．

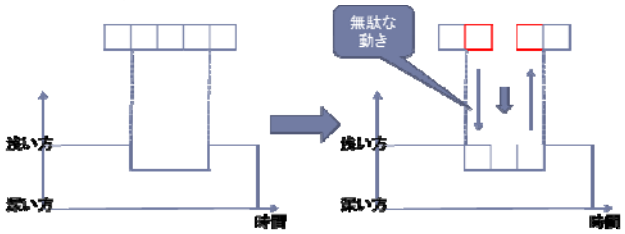

Fig. 8 Splitting the stock period of an unassigned block.

4. シミュレーション実験

本実験では 264 日間にストックヤードへ蔵置されるブロックの計画データを用いて，Fig.2 で示されるストックヤードに蔵置するスケジューリングを行った．ストックヤードに蔵置されるブロック数は合計 3061 個になり，またこの造船所ではストックヤードの能力限界よりもブロック搭載日程の遵守のほうが優先事項であったため，日によってはストックヤードの最大蔵置能力（109 個）を 20 個程度上回る場合もある．

Table 1 Results of the simulation.

Method	Number of unassigned blocks	Total number of unassigned block space
Branch-and-Bound 1 (300branches)	139	1030
Branch-and-Bound 1 (600 branches)	149	1058
Branch-and-Bound 2 (500branches)	213	1126
Branch-and-Bound 2 (1000branches)	213	1118
Branch-and-Bound 1 (300branches) and Splitting stock periods of unassigned blocks	192	629

Table 1 は各手法を用いてスケジュールを行った結果得られた解の質を，割当てられずに残ったブロック数（Number of unassigned blocks）および割り当てられずに残ったブロックの蔵置期間の合計（Total number of unassigned block space）で評価した様子を示す．ここで，深さの深いスタックから優先的に埋めていくというヒューリスティクスが意外と強力であることが分かる．また，一般に分枝限定法の枝数が多いほうが，見つかる最良解の質が改善されるが，上記のヒューリスティクスと組み合わせた場合はそうでもないようである．分枝限定法 1 の枝数 300 で探索して得た解が最も良かったので，この解に 3.3 節の「配置できなかったブロックの分割蔵置」をスケジュールした結果が Table 1 の最下段で，本実験で見つかった最良解では，延べ 629 個のブロック蔵置期間が未割当となった．しかしこれは先にも述べたとおりストックヤードの最大蔵置能力を超えてブロックが入ってくることが大きな要因である．Fig.9 の赤線はストックヤードで蔵置しているブロック数の変化を表す．この赤線が左側の数字 109 の部分を超える期間では，蔵置し切れないブロックが必ず発生する．Fig.9 の緑線は，上記の延べ 629 個のブロック蔵置期間が未割当のスケジュールに従った場合の未割当ブロックの個数の変化を表す．このグラフより，ブロックの蔵置個数がストックヤードの蔵置限界を超えていないのに置ききれないブロックが発生している箇所がごくまれに発生しているが，ほとんどの期間では異常無く，良好なスケジュールを自動生成できたと判断できる．

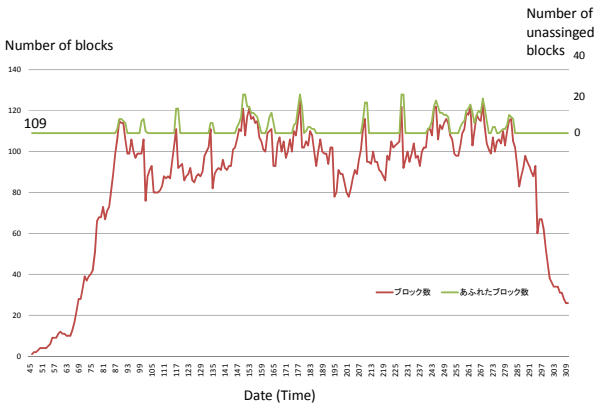

Fig. 9 The number of blocks and the number of unassigned blocks in the block stockyard.

5. 考　　察

関連研究としては，スタック構造で蔵置する物流システムとしてコンテナヤードが挙げられ，研究例[2]が多いが，コンテナの出入りが確率的だったり，置場にもっと余裕があるなど相違点が多く，利用は困難である．

本システムの実用化へ向けての課題として 1）途中から再スケジュールを行う機能，2）並行部ブロック置場と曲がり部ブロック置場の区別，などが挙げられる．

謝　　辞

本研究を遂行するにあたりまして，株式会社大島造船所の人位康弘様より貴重なデータおよびコメントをご提供いただきました．ここに厚く御礼申し上げます．また本論文の図表は斉藤正幸君の修士論文[3]を利用しました．

参 考 文 献

1) 田房友典，井手浩文：造船組立てにおけるブロック配置システムの開発，日本船舶海洋工学会講演会論文集，第 11 号, 2010, pp.463-466.
2) 樋口良之，阿部雅二郎，伊藤廣：ファジイ理論を用いたコンテナターミナルの最適蔵置法，日本機械学会論文集(C 編)65 巻 634 号(1996-6)，pp.431-436.
3) 斉藤正幸：造船大組ブロックストックヤードの物流に関する研究，平成 22 年度九州大学大学院修士論文