物流倉庫における荷揃え業務とパッキングの最適化

2013SE042 本田章太郎　2013SE118 宮嶋綾
指導教員：佐々木美裕

1　はじめに

倉庫業は, 生産と消費を結ぶ産業として国民生活の基盤を支える極めて公共性の高いものである. 倉庫では物品の特性に合わせた保管をはじめ, 検品, 入庫, 流通加工, ピッキング, 仕分け・荷揃え, 出庫など, 様々なサービスが提供されている.

ある会社ではメーカーより寄託を受けたアイスクリームや冷凍食品に入出荷・保管業務を行っており, 問屋や配送センター向けの共同配送業者への出荷・引渡し業務を冷蔵倉庫にて行っている. この出荷・引渡し業務はトラックで行われ, そのトラックを配送車両と呼ぶ. この冷蔵倉庫は 4 階建てで, 2 階から 4 階はメーカーから寄託を受けた商品を保管している商品棚, 1 階は仮保管庫となっている. 仮保管庫は, その日に出荷引渡し業務が行われる商品を保管する場所である. 1 階にある仮保管庫に一時的に保管しておくことにより, 配送車両への積み込み業務を円滑に行うことができる. また, その際に商品の状態が悪くならないよう商品を低温で保管するなどの役割がある.

図 1 は, 作業の流れを表している. 2 階から 4 階の作業担当者は, ピッキングリストに従って商品のピッキングを行う. ピッキングリストには, ピッキングする商品とともに出荷時に積み込む配送車両とその出荷時刻が記載されている. 同じ配送車両に積み込む商品をまとめたものをパックと呼び, パックはパレットと呼ばれる荷役台の上に乗せられて (図 2 参照) エレベーターなどで 1 階に運ばれる. 1 階では, 同じ配送車両に積み込むパックを積み重ねる作業を行う. ここで, パックを積み重ねたものをユニットと呼ぶことにする (図 2 参照). ピッキングされた商品の種類や量によってパックの高さは, さまざまである. もともと高さのあるパックの場合, そのままひとつのパックをひとつのユニットとすることもある. 各ユニットは出荷時刻までの間, 仮保管庫にて保管され, 出荷時刻前になると仮保管庫から出庫されて配送車両に積み込まれる. ユニットを保管するために, 仮保管庫に詰め込む作業をユニットのパッキングと呼ぶことにする.

これらの作業は, 担当者の経験と勘で行われている. 作業担当者は冷凍食品が保管されている冷蔵倉庫で, これらの作業を行わなければならない. そのため これらの作業時間を短縮することができれば, 担当者の負担を減らし, 円滑に作業することができるのではないかと考え, 本研究ではユニット作成数の最小化と仮保管庫へのパッキングの効率的な作業方法を考える.

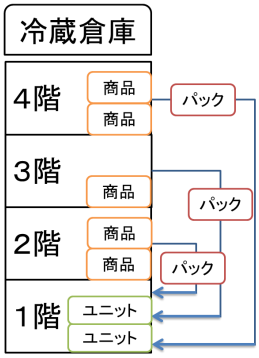

図 1　作業の流れ

2　ユニット作成とパッキング

2.1　ユニット作成について

1 階の仮保管庫はパックを十分に並べるスペースがないため, 配送車両ごとにユニットとしてまとめる必要がある. パックの大きさは, ピッキングされた商品により異なり, ユニット作成の際はパレットも一緒に積み重ね, 下のパックが上のパックの重みで潰れてしまわないようにする. また倉庫の入口の高さにより, ユニットの高さには制限がある.

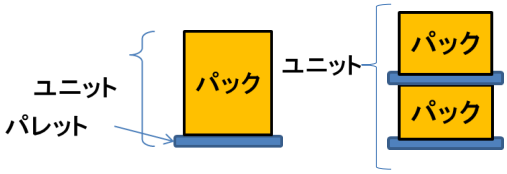

図 2　パック, ユニットの構造

2.2　パッキングについて

パッキングは, ユニットを配送車両に積み込むまでの間, 一時的に 1 階の仮保管庫に詰め込み商品の状態を保つために行う. 仮保管庫は出入口がひとつの先入れ後出し構造である. 商品は冷凍食品であるため, ユニット作成で完成したユニットから保管庫に詰め込む. パッキングの際のユニットの運搬には, フォークリフトを使用する. このとき, パレットの穴の部分にフォークリフトのフォーク (ツメ) を差し込み持ち上げて運搬する. 仮保管庫のスペースとパ

レットの構造上からユニットは一方向にしか運搬することができない．そのため, 早く出発する配送車両に載せるユニットは仮保管庫の手前 (入口) 側に, 遅く出発する配送車両に載せるユニットは仮保管庫の奥側に詰め込まなければならない．

3　ユニット作成数最小化モデル

ユニットの高さ上限を詰められる重さの上限とみなし, 時刻制約を加えることで, ユニット作成問題は時刻制約のあるビンパッキング問題として定式化できる．

3.1　記号の定義

この問題を解くにあたり, 使用する記号を以下のように定義する．

添字集合

P :　パックの集合
U :　ユニットの集合
c :　ユニットの高さの上限
m :　時刻
h_i :　パック $i \in P$ の高さ
s_i :　パック $i \in P$ が各階でピッキングされ, 1 階に到着する時刻
a_k :　ユニット $k \in U$ の重み
($a_1 < a_2 < \cdots < a_{|U|}$ を満たすように設定)
$R_i = \{j \in P \mid |s_j - s_i| \geq m\}$, $i \in P$: パック $i \in P$ が 1 階に到着する時刻との到着時刻の差が m 以上あるパック

決定変数

$$x_{ik} = \begin{cases} 1 : \text{パック } i \text{ をユニット } k \text{ に入れる} \\ 0 : \text{パック } i \text{ をユニット } k \text{ に入れない} \end{cases}$$

$$y_k = \begin{cases} 1 : \text{ユニット } k \text{ を使用する} \\ 0 : \text{ユニット } k \text{ を使用しない} \end{cases}$$

3.2　定式化

上記の記号を用いて以下のように定式化する

$$\text{min.} \quad \sum_{k \in U} a_k y_k \tag{1}$$

$$\text{s.t.} \quad \sum_{k \in U} x_{ik} = 1, \qquad i \in P \tag{2}$$

$$\sum_{i \in P} h_i x_{ik} \leq c y_k, \qquad k \in U \tag{3}$$

$$x_{ik} + x_{jk} \leq 1, \quad i \in P, j \in R_i, k \in U \tag{4}$$

$$x_{ik} \in \{0, 1\}, \qquad i \in P, k \in U \tag{5}$$

$$y_k \in \{0, 1\}, \qquad k \in U \tag{6}$$

式 (1) は, 使用するユニット数の合計を最小化する目的関数である. また, a_k を使用することで同じ高さのユニット同士での区別をし, 計算時間の短縮を図る. 式 (2) は, すべてのパック $i \in P$ をいずれかのユニット k に含む制約である. 式 (3) は, パック $i \in P$ を積むユニット k の高さの制約である. 式 (4) は, パック $i \in P$ とパック $j \in P$ において, 1 階に到着する時刻が m より小さければ同じユニットに積み重ねることができるという制約である. 式 (5), (6) はバイナリ制約である.

4　ユニット作成数最小化のアルゴリズム

本研究では, ビンパッキング問題の近似解法として, FF 法と BF 法のアルゴリズムを用いる.

4.1　ファーストフィット法 (FF 法)

荷物を箱の添字番号が若い順に詰めていく．このとき, 荷物を詰めると, 箱に詰められる荷物の重さの上限を超えてしまうなら, その箱を閉じて新たな箱を用意するという解法である [1].

4.2　ベストフィット法 (BF 法)

BF 法とは, FF 法と同様に箱に詰めていくが, 荷物を添字番号が若い箱ではなく最も中身の詰まった箱に入れる解法である [1].

5　パッキング

5.1　詰め込み問題

詰め込み問題とは, 与えられた図形をある大きさの容器に図形の重複がないように配置していく問題である [2]. 図 3 は仮保管庫を上から見た形になっている．上から見たときのユニットはパレットの大きさになるものとし, またすべてのパレットの大きさは縦横の長さが同じ正方形であるとする．仮保管庫内に行と列を設定し, 左奥を 1 行 1 列とする．仮保管庫内のひとつのマスにはユニットがひとつ保管できるスペースがある．これにより, パッキングは, 仮保管庫を上からみることで, 仮保管庫を大きな容器, ユニットを与えられた図形として, 2 次元の正方形の詰め込み問題といえる．また仮保管庫のスペース, パレットの構造とフォークリフトの使用によりユニットは仮保管庫の奥から手前, 手前から奥へ向かう一方向でしか運搬することができない.

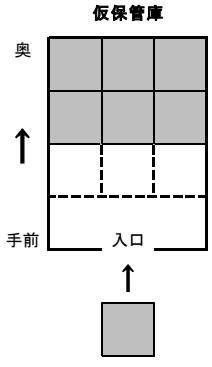

図 3　仮保管庫のレイアウト

5.2 モデル化

ユニットを仮保管庫の奥から詰め込むときの最適配置を貪欲算法で考える. このとき考えなければならない制約は, ユニットの前後関係である. ユニットは, 入庫時刻と出庫時刻をもっている. 入庫時刻とはユニットが完成して仮保管庫に詰め込む時刻のことをいい, 出庫時刻とはユニットを配送車両に載せる時刻のことをいう.

時刻の制約は 2 つある. ひとつはユニットを保管する位置の奥にユニットが保管されている場合, 保管するユニットの入庫時刻は奥に保管されているユニットの入庫時刻よりも遅いか同時でなければ, その位置にユニットを保管することができない制約である. これは完成したユニットからつぎつぎと仮保管庫に保管しなければ, 2 階から 4 階の各階からパックが送られ 1 階の作業場が混雑するためである.

もうひとつはユニットを保管する位置の奥にユニットが保管されている場合, 保管するユニットの出庫時刻は奥に保管されているユニットの出庫時刻よりも早いか同時でなければ, その位置にユニットを保管することができない制約である. これは, 仮保管庫が先入れ後出し構造であり, ユニットの運搬が奥から手前, 手前から奥の 1 方向のみでしか行えないためである.

5.3 記号の定義

本研究では, ユニットを左奥から優先的にパッキングを行うアルゴリズムを考える.
アルゴリズムを記述するにあたり, 以下の記号を定義する.

I : 仮保管庫の行数
L : 仮保管庫の列数
K : ユニットの集合
U : 保管場所が決定したユニットの集合
$\bar{U}$: 保管場所が決定していないユニットの集合
D : 保管することができなかったユニットの集合
e_u : ユニット $u \in K$ の入庫時刻
l_u : ユニット $u \in K$ の出庫時刻
m_{ij} : 仮保管庫の i 行 j 列の位置に保管してあるユニット

[仮保管庫へのパッキングのアルゴリズム]

1. U:=ϕ, $\bar{U}$:=K, m_{ij}=0.
2. E =argmin$_{u\in\bar{U}}\{e_u\}$, $\hat{u}$ =argmax$_{u\in E}\{l_u\}$. m_{ij}=0 のとき, 5 へ進む. m_{ij}>0 ならば, 6 へ進む.
3. i=0 のとき, 5 へ進む. $i\neq$0 のとき, 4 へ進む
4. $e_{\hat{u}}\leq e_{m_{i-1j}}$, $l_{\hat{u}}\geq e_{m_{i-1j}}$ ふたつの制約を満たすとき, 5 へ進む. 満たさないとき, 6 へ進む.
5. m_{ij}=$\hat{u}$, $\bar{U}$:=$\bar{U}\backslash\{\hat{u}\}$, U:=$U\cup\{\hat{u}\}$, i=0, j=0. 7 へ進む.
6. $i \neq I$ のとき $i := i+1$ とする.$i = I$ のとき $j := j+1$, $i := 0$ とする. $i = I$ かつ $j = J$ のとき $\bar{U}$:=$\bar{U}\backslash\{\hat{u}\}$, D:=$D\cup\{\hat{u}\}$ となる.7 へ進む.
7. $\bar{U}$=ϕ のとき, 終了する.$\bar{U}\neq\phi$ のとき, 2 へ戻る.

5.4 パッキングの例

図 4, 5 を例として示す. この図は, 保管スペースが 3 行 3 列の仮保管庫であり, ユニットをパッキングした状態である. 仮保管庫を上から見た形になっており, 上が奥で下が手前である. 数字がついている正方形はユニットであり, 何も書かれていない正方形はユニットが保管されていないスペースである.

図 4 は入庫時刻が 3 までユニットをパッキングした仮保管庫である. 数字は入庫時刻であり, 1 から順に入庫時刻が早いユニットである. 入庫時刻の 4 のユニットをパッキングする. このとき, 左奥から優先的にパッキングを行うため, 最初に 3 行 1 列の座標でパッキングを考える. この位置の奥, 2 行 1 列のユニットの入庫時刻が 2 であるため, 入庫時刻が 4 のユニットは 3 行 1 列の位置にパッキングする.

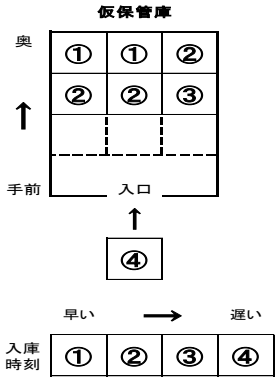

図 4 入庫時刻を考慮したパッキング

図 5 は出庫時刻が 4 までユニットをパッキングした仮保管庫である. 数字は出庫時刻であり, 数値が小さい方が出庫時刻が早いことを示す. 出庫時刻の 2 のユニットをパッキングする. このとき, 左奥から優先的にパッキングを行うため, 最初に 3 行 1 列の位置でパッキングを考える. この位置の奥, 2 行 1 列のユニットの出庫時刻が 1 であるため, この位置にはパッキングすることはできない. よって次の列の位置 3 行 2 列でパッキングを考える. この位置の奥, 2 行 2 列のユニットの出庫時刻が 2 なので出庫時刻が 2 のユニットは 3 行 2 列の位置にパッキングする.

6 計算結果と考察

6.1 データの作成

本研究では, 各階で作業員がピッキングした, パックが 1 階へ到着する時刻を考慮したユニット作成をそれぞれの解法 (ビンパッキング問題, FF 法, BF 法) で考える. このとき, 積み込まれる配送車両ごとに計算を行う.

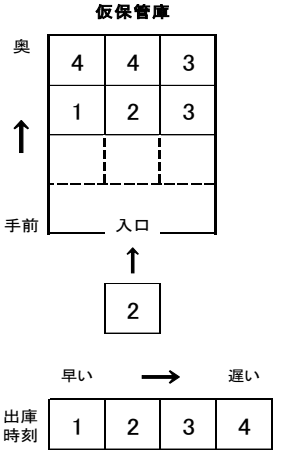

図 5 出庫時刻を考慮したパッキング

計算には, パックの 1 階への到着時刻が全て同じ (全てのパックが出揃った状態と同義), 1 階への到着時刻が 1 から 10 の時刻でランダムとなっているデータを用いる. パックの高さは, 1 から 5 の高さでランダムになっている. この会社では, 通常期には 300 個から 1000 個のパック, 繁忙期には 600 個から 2000 個のパックが作られ, 各配送車両に積み込まれるため, 配送車両 1 台あたりでは, 100 個から 200 個のパック数となると考えられる. そのため, パックの数を 100 個, 200 個として実験を行う.

それぞれの解法で作成されたユニット数, 最適化にかかった計算時間を比較する. ユニット作成で得られたユニットのデータを用いて, パッキングの最適化を行い, 計算結果を考察する.

6.2 実行結果

最適解を求めるために使用したソフトウェアは, Gurobi6.5.2 であり, 計算環境は (プロセッサ: Intel(R)Core(TM) i5-3320M CPU @ 2.60Ghz 実装メモリ: 8GB) である. BF 法, FF 法は, C 言語で実装した. 計算環境は (CPU: Intel(R)Core(TM) i5-3320M CPU @ 2.60Ghz OS: Linux version 4.6.3 メモリ: 7.8GiB コンパイラ: gcc version 4.6.3) である.

Gurobi と FF 法, BF 法の計算結果を表 1, 表 2 に示す. パック数が 100 個では, 解法ごとに求められるユニット数には差があまり見られなかった. しかし, パック数を 200 個として計算すると, FF 法, BF 法は計算時間がほとんどかからなかったため, 表 6.2 と比べると圧倒的に早いことが分かる. また, パック数が極端に大きくなった場合, Gurobi では, out of memory で計算不可となることがあるが, FF 法, BF 法では解を出すことができた. ユニット作成数が, BF 法と FF 法では多くなってしまうことがある. ユニットに含まれるパックを見ると, ユニット毎の完成時間に大きな見られ, BF 法, FF 法で解いた場合のほうが, それぞれのユニットの完成時間の平均が早くなることが多くなった.

2 つのパックの到着時刻の差	Gurobi	FF 法	BF 法
2	67	70	70
3	67	70	70
4	67	70	70
10	67	70	70

表 1 ユニット作成数 (パック数=100)

2 つのパックの到着時刻の差	Gurobi	FF 法	BF 法
2	122	124	124
3	121	123	123
4	119	123	123
10	119	123	123

表 2 ユニット作成数 (パック数=200)

2 つのパックの到着時刻の差	パック数 100 個	パック数 200 個
2	12.8	574
3	11.0	237
4	9.03	212
5	5.38	172

表 3 Gurobi の計算時間 [s]

パッキングをユニット作成で求められたデータを用いて計算した結果, パッキング出来ないユニットが出てきてしまった.

7 おわりに

パッキングが出来ない原因として, 入庫時刻が早いユニットを奥から詰めていくため, 入庫時刻が早く, 出庫時刻が遅いユニットが先にパッキングされてしまうと, それよりも出庫時間が遅いユニットがパッキング出来ず溢れてしまっていると考えた.

FF 法, BF 法において, パックをユニットに積み重ねる順番が変わることでユニット作成数, 完成時間が変化すると考えられるため, 1 階への到着時刻が同じパックの順番を並び替えることで, 今回の実験とは異なる結果が得られ, 最適解に近づくことができると考えた. 計算結果よりパック数が多くなればなるほど, FF 法, BF 法が適していると考えられる. また, ユニット作成の最適化は, パッキングの最適化に繋がると考えた.

参考文献

[1] 藤澤克樹, 梅谷俊治 :『応用に役立つ 50 の最適化問題』. 朝倉書店, 2009.

[2] 野澤貴博 : 詰め込み問題における近似アルゴリズムの研究, 法政大学大学院デザイン工学研究科紀要, Vol.3, p.1-2, 2014.