物流システムにおける空間の最適利用化エキスパートシステム*

福田 敏 男^{*1}, 永 田 恭 一^{*2}

Expert System for Packing Space Utilization in a Material Flow System

Toshio FUKUDA and Kyoichi NAGATA

This paper describes an allocation and distribution planning method for packing objects in a material flow system. We consider a problem of packing objects in an allowable space in this paper, for which the knowledge-based system is proposed with the space geometrical expression methods, assuming that objects are rectangular. In packing objects in the admissible space, the space left for the allocation is divided into several subspaces, according to the size of the objects. Then, these divided subspaces and objects are represented by the relation of connection and the division of the space. For the reasoning method, we employ human like heuristic knowledge: in cases in which the width of an object is larger than that of the allowable space left, we cannot pack the object in the space, but can pack the object by applying the force and pushing the object into the space. In cases of packing the object in unfavorable space, the system takes two steps backwards and reconsiders the packing process in a better way by the backtruck mechanism of the system. This paper shows some comparisons of packing efficiency for the evaluation of the knowledge-based system with the basic rules, so rules, so that the packing planning system can be improved by different additive rules, so as to decrease the dead space in the total area. Extensive simulations have been carried out to evaluate different system performances.

Key Words: Artificial Intelligence, Production Planning, System, Material Flow, Planning Expert System, Space Allocation

1. は じ め に

物流システムの近代化のためのシステムの自動化が急がれている。この内、物流システムにおいて物体を箱に積めるハンドリングの際に必要なことは、残っている空間における物体の配置プランニングを行うことである。従来の物流システムの箱詰め作業の配置プランニングでは、形状が既知の物体を対象としているために、組合せによる方法、簡単な規則を利用する方法等によって最適な解を見つけたことができた。また、空間の表現方法として、空間を座標点で表し、その座標点における物体の有無を1と0で表すことが多く、本論文のように幾何学的形状で表現することが少なかった。本研究においては二次元平面空間を対象として知識ベースシステムを用いる⁽²⁾⁽³⁾。また、物体を積めるごとにすべての空間をく形に分割し表現し直した新しい空間はその隣接空間および物体に関するすべての相対位置関係で表現する。また、本研究においては次々に送られてくる物体はく形ということ以外わからない状態を想定して行うため、物体の配置表現方法と

して、人間の考えるような経験や感のようなあいまいさはあるが、ある程度適当な配置ができるような知識を取り入れ、ルールとして用いる。また空間の幅が物体の幅よりも狭い場合に物体を押し込むことによる詰め込みを行い、今までできなかった物体に力を加えたときの変形について考察する。またルールの違いによる物体の詰め方、必ずしも人間が考えるような最適でない空間を積めた場合の詰め替え動作(バック・トラック)を行ったときの効果について考察することを目的とする。

2. 特徴および仮定

本研究を行うにあたり、三つの特徴を与えた。

- a. 空間、物体をそれぞれの幾何学的位置関係で表現。
- b. 物体に外見以外の物理的(強さ、硬さ、その他)情報を与える。
- c. 力を加えたときの物体の力学変形を考慮。

空間、物体をそれぞれの幾何学的位置関係で表現するために物体の形状は限定されてしまう。また、本システムの有効性の確認、簡略化のため本研究ではこれらを三つの仮定として与えた。

- a. 物体をく形とする。
- b. 空間を二次元平面とする。

* 原稿受付 平成2年3月30日。

^{*1} 正員、名古屋大学工学部 (〒464-01 名古屋市中千種区不老町1)。

^{*2} 東京理科大学大学院〔現在 日本電気〕。

c. 初期空間として3方が壁で囲まれた、正方形の空間を用いる。

3. 空間と物体の表現方法

3・1 空間の生成と位置表現 空間をどのように表現するかにより、二次元平面空間における物体の詰め方が変化する。従来行われていたように、空間をドット化しそこに物体が存在するか存在しないかを2進数の0と1で表現した場合、そのドットの数だけデータが増えてしまうため、それだけ情報量が増えてしまう。また本研究で用いたプログラミング言語(Prolog)でこのような膨大な量で、また座標としか表現されないデータを表現することはメモリ容量、処理時間において限界がある⁽⁴⁾。そこで新しい空間の表現の方法を提案する。く形の空間にく形物を置くと、残った空間を新たに二つ以下の空間として表すことができる。したがって、空間についてもく形物の集まりであると仮定すれば、物体と空間の間の位置関係をそれぞれの隣接関係により容易に判断できる。図1(a)のように物体C,D,空間X,Yが存在するときに物体Aを空間Yに積めた場合、空間Yは図1(b)のように空間Y1,Y2になる。

3・2 物体、空間の位置表現 物体および空間の位置を表現するためには、物体の座標を考える必要がある。しかし、常に座標を用いて表現するのは効果的でない。そこで必要なとき以外はそれぞれの隣接関係

を用いて表現する。隣接する物体、空間の直接結び付いているという関係をエキスパート・システムの知識表現のうちの意味ネット・ワークに以て表現方法を用いて表す。

図2(a)のように物体A,B,C,D,空間X,Y,Zが存在するときの物体Aの位置は、Prologの表現を用いて⁽¹⁾⁽²⁾、左右方向に関して

$\text{right}(a,c) \dots\dots\dots A \text{ は } C \text{ の右側}$

$\text{right}(b,a) \dots\dots\dots B \text{ は } A \text{ の右側}$

$\text{right}(z,a) \dots\dots\dots Z \text{ は } A \text{ の右側}$

のように表現する。また内側、外側の関係を

$\text{inner}(a,y) \dots\dots\dots A \text{ は } Y \text{ の内側}$

$\text{inner}(d,a) \dots\dots\dots D \text{ は } A \text{ の内側}$

のように表現する。また

$\text{left}(X,Y) : \neg \text{right}(Y,X)$

という述語を用いれば

$\text{left}(a,b) \dots\dots\dots a \text{ は } b \text{ の左側}$

といった表現も可能になる。

また、XとAのように接していない物体関係、または物体と空間の関係を、XとAについて、

$\text{inner}(c,x), \text{right}(a,c)$

のように物体Cを使った関係として表すことができる。

したがって、BとCとの関係を

$\text{right}(b,a), \text{left}(c,a)$

(a) 空間の生成前

(b) 空間の生成後

図 1

(a) 空間、物体の位置表現

(b) 空間、物体間の関係

図 2

$=\text{right}(b,a), \text{right}(a,c)$

と共通の述語に共通の因数 A が現れたので

$\text{righthand}(b,c) \dots \dots B$ は C の右の方

という述語を定義し表現できる。これらの隣接関係を図2-2に示す。したがって、ある物体または空間を指定したとき、基準となる場所との位置関係を表現すれば、その位置の座標も表現できる。

3.3 空間および物体の有する情報とその表現
空間はその生成された順番にアドレスをつけられる。したがって、ある状態で存在するアドレスは1, 2, 3, ..., とはならない。したがって、その状態ですべてのアドレス(1, 2, 3, ...)が存在するとは限らず、過去の状態で使用したアドレスは二度と使えないものとする。これは過去の状態での同じアドレスをもつ空間との重複を避けるためである。よって、隣合う空間を捜すにはアドレスが1違いの空間を捜すのではなくデータ・ベースより位置関係で探索する。また、空間はく形であることから、高さと幅をもち次のように表す。

$\text{space}(N, [X_s, Y_s])$

X_s : 高さ

Y_s : 幅

N : 空間のアドレス

物体の場合、送られてきた順番でアドレスをつける。したがって、物体のアドレスとその物体を詰めたときの状況とが一致するため過去の状況と呼び出すのに有

効となる。また物体 Z という項を持つ。本研究ではこれに物体の物理的な情報である物体の硬さと強さを与えているが、システムの向上に伴い物理的情報以外の情報を持たせることも可能である。

$\text{object}(I[X_o, Y_o], [Z])$

X_o : 高さ

Y_o : 幅

I : 物体の詰められた順番

Z : 物体のもつ情報

4. 推 論 方 法

4.1 空間の結合 物体を詰めていくと、空間の数が増えていき、また空間の幅は狭くなっていく。物体と空間のマッチングは1対1であるため空間が小さくなると物体が詰められなくなる。したがって、空間の幅を広くするために隣合う空間同士を結合し新しく空間を生成し直す必要がある。図3のような空間 S_1 , S_2 を結合する。新しい空間の幅は $Y_{s1} + Y_{s2}$ となり高さは $\min(X_{s1}, X_{s2})$ となる。ここで現れた D は dead space と呼び物体を詰めることのできない空間と定義する。理由として dead space は他の空間に比べ一般的に面が小さい場合が多く、この空間に物体を詰められるかどうかの探索を行うことは不必要だからである。しかし一度 dead space が増えてくると、詰め込む効率が悪くなる。また、dead space でなくても幅の狭い空間が存在するだけで、次から詰める物体の傷害ともなり得る。したがって隣合う空間をただ結合するのではなく、適当な隣合う空間を選び結合する必要がある。結合する空間を選択するためにルール・ベースを用いる⁽²⁾。このルール・ベースを大きく三つのルール群に分けた。

①物体とマッチングする空間が存在しない(物体を詰められる空間がない)。

②空間(の幅)が小さくなった場合の隣接する空間への吸収。

③ dead space を再び普通の空間に戻す。

①は結合することにより物体をうまく詰めることができるような空間にする必要がある。また、詰められる空間が存在しても、そこが適当でない場合にはこのルールを適用する必要がある。②ではどのくらい狭い幅であるかを認識するためのルールをもつ必要がある。図4のように空間 S が物体 A, B に囲まれ、 S の幅が狭く物体を詰めることができないと判断された場合、 S は空間 S_r に吸収される。したがって A, B に囲まれた部分が dead space となるためこの面積が小さいかを判断する必要がある。③は一度 dead space となっ

図3 空間の結合

た空間を利用できるようにするために必要である。図5のような形で物体、空間、dead spaceが存在した場合、結合を行う。①, ②, ③で共通するルールとして、できるだけdead spaceを小さくするように空間の結合を行う必要がある。

4.2 空間の選択 物体を積める際、一つずつベルトコンベアなどにより送られてくる場合、ある程度の経験が必要になってくる。実際、物体を積めるのは、人によって違ってくるが、それでも最終的には最適とはいえないまでも、ある程度満足できる積み方ができる。本研究では次に送られてくる物体はく形ということ以外なにもわからない状態で行う。また送られてきた物体は必ず空間のどこかに詰めることを条件とする。

空間を選択するとき必要な条件を、人間の感に基づいたルール・ベースを用いて記述する。推論の方法とし

図4 small spaceの結合

図5 dead spaceの吸収

図6 ルールのアルゴリズム

てマッチングしたすべての空間について、物体を詰めたと仮定したときのその隣接した情報より評価する。評価の方法は各ルールとのマッチングを行い成功した場合にその空間の評価関数を増す。本論文ではルール・ベースを用いて推論を行っているため、推論の際にそれぞれのルールに重みをつけており、マッチングしたルールの重みの合計を評価関数と定義した。また、各ルールの重みは試行錯誤し決めた値ですべての重みは1~10の範囲で表現した。そして、最大の評価を得た空間に物体を積める。物体を詰め込む空間は、人間の考える空間と同一とし、これを最適空間とした。基本ルール・ベースの例を用いた場合のルールの表現方法を図6に示す。例としてマッチングする空間が存在しない場合、また存在してもその空間の左右に空間しか存在しない場合に空間の結合を行う。空間の選択では、物体を詰める空間につけて詰める評価を増し、また空間の左側の他の物体が存在するとき物体を空間の左側につけて詰める評価を増す。これらの推論とデータ・ベースの更新の実行例を図7から図9に示す。また、本システムのフローチャートを図10に示す。

if put obj(350,200) in sp32,
sp21 is useless,
then unite sp32 and sp21

Input object : object(14, [350,200]).

マッチングしたスペースは
[space(32, [500,200])]
オブジェクトを置けるスペースは
[space(32, [500,200])]
オブジェクトを置けるスペースはありません
space(32, [500,200])とspace(21, [480,110])を結合します
space(33, [480,310])を追加します
d_space(34, [20,200])を追加します
space(32, [500,200]) space(21, [480,110])に関する情報を削除します

図7 空間の選択及び空間の結合

put obj14 in new sp33,
and add new sp35

マッチングしたスペースは
[space(33, [480,310])]
オブジェクトを置けるスペースは
[space(33, [480,310])]
object(14, [350,200])をspace(33, [480,310])の右側に付けて置きます
space(35, [480,110])を追加します
space(36, [130,200])を追加します
space(33, [480,310])に関する情報を削除します

yes

図8 物体の詰め込みとデータ・ベースの更新

5. 力による物体の変形

5.1 力情報に関する変形 物体に弾性度を持たせることによって、その左右に力が加わったとき変形を考える。一つ一つの物体は固有の性質を持っているため、特徴をすべて表現するのは大変である。したがって本研究は三つの仮定を用いて行った。

- ①物体をばねとして表現する。
- ②力を加えたときの横ひずみを無視する。
- ③物体は力の方向にゆがむ。

以上の仮定により、物体をばね系にモデル化できる。

5.2 力情報の近以モデル化および実行結果 ばねとして表現された物体はその物体のゆがむ割合と物

体にかけられる最大の力により、力を加えられ変形するがこれを解くには力の加わるすべてのばねの数と同数の変数を持つ方程式を解く必要がある。そこで近以モデルを利用することを考える。理由として第一に物体の数がいくつになっても、方程式を解かずに最大応力およびばね定数が表現できるためである。第二に物体間の表現方法を利用できるためである。図11を用いて近似モデル化の方法を述べる。図11の近似数学モデルを示すと図12のようになる。図13は図11のモデルに一般性をもたせたモデルである。物体を詰め込むべき空間から左右の壁への道のりを探索し、一つの探索路が一つの直列ばねを形成すると考える(図

Input object : object (15, [400, 100]).

space (35, [480, 110]) と d_space (34, [20, 200]) を結合します
space (37, [500, 110]) を追加します
d_space (38, [20, 90]) を追加します
space (35, [480, 110]) d_space (34, [20, 200]) に関する情報を削除します
マッチングしたスペースは:
[space (37, [500, 110])]
オブジェクトを置くスペースは
[space (37, [500, 110])]
object (15, [400, 100]) を space (37, [500, 110]) の右側に付けて置きます
space (39, [500, 101]) を追加します
space (40, [100, 100]) を追加します
space (37, [500, 110]) に関する情報を削除します

図9 dead spaceの再利用

図10 システムのフローチャート

1. {C, A}
2. {B, A}
3. {G, F, E}
4. {G, D}

図11 近似モデル例

図12 モデルのばね化

図13 各ばねの合成ばね化

12). この場合、左右で二つずつのばねがあると考え
る。しかし物体 A,G は重複されているため、他の空間
よりも大きな力が加わっていると仮定しそれぞれの弾
性度、強度を重複回数で割った大きさで考える。また、
物体の重複がない場合には、近似モデル値が理論値と
一致する。

図 13 の合成ばね定数を求め図 14, 図 15 により、こ
の近似モデルの有効性を確かめる。図 13 に示すモデル
の合成ばね定数については、物体のばね定数がハンド
リングするごとに異なるため、物体の物体の取り得る
範囲のランダムな値を与えている。合成ばね定数その
ものは、合成するごとに異なるが、Prolog 上で行っ
た簡単な計算結果である近似解と、別に計算した理論
解との誤差について比較する。E 1 ~ E 6 は、図 13 に
示すモデルにランダムな値を代入して 6 回計算した結
果を示している。また K 1 ~ K 5 はランダムな値と
して取扱った。この結果より近似解に 20% の余裕を持
たせて考えると、この近似モデルを利用できることが
わかる。20% という値は本論文での近似モデルを用い
てシミュレーションした結果、ばね定数がすべての場
合において理論解との誤差が 20% 以内におさまった

	Elasticity	Approximation	error(%)
E 1	62.5	58.3	6.7
E 2	65.5	52.2	19.3
E 3	72.9	62.6	14.7
E 4	72.2	59.5	17.6
E 5	84.4	77.4	8.3
E 6	133.3	111.7	16.2

図 14 合成ばねの理論値と近似値

図 15 実行回数と合成ばね定数の関係

ため用いた。

例として、図 16 の S 14 に力を加えて物体を詰め込
んだときの結果を図 17 に示す。幅 200 の空間に最大応
力 1000、弾力度 100 の物体を詰め込むと仮定したと
き、空間の広がった幅と物体の縮んだ幅の合計は 17.8

```
! ?-listing(space).
space(7, [670, 100]).
space(8, [680, 120]).
space(12, [760, 130]).
space(13, [800, 180]).
space(14, [1000, 200]).
space(15, [700, 270]).
yes
! ?-listing(d_space).
d_space(16, [20, 150]).
yes
```

図 16 実行結果例

```
?- start.
wide, force and elasticity of the object
wide 215.
force 1000.
elasticity 100.
```

```
minimum force of all object is 300.0
number of ways to the right wall are 3
number of ways to the left wall are 2
elasticity of right objects is 70.0
elasticity of left objects is 28.57142857
shrink length is 17.78571429
wide of the space is 217.78571429
the wide is just, and put object in the space
yes
```

(a) 幅が 215 のとき

```
?- start.
wide, force and elasticity of the object
wide 220.
force 1000.
elasticity 100.
```

```
minimum force of all object is 300.0
number of ways to the right wall are 3
number of ways to the left wall are 2
elasticity of right objects is 70.0
elasticity of left objects is 28.57142857
shrink length is 17.78571429
wide of the space is 217.78571429
the wide is short
yes
```

(b) 幅が 220 のとき

図 17 実行結果

図 18 各システムのルール群

図 19 各システムのルール数

となる。したがって、幅 215 の物体は詰め込めるが、幅 220 の物体は詰め込めない。空間に物体が押し込まれたときのデータ・ベースは、空間と物体の幅が一致したと表現される。したがって、物体の幅が 200~217.8 で空間と幅が一致するため他の空間に物体を詰めるよりも効率よく詰められる。

6. バック・トラックによるシステムの向上

残りの空間が少なくなったとき大きな物体が送られてきた場合、詰めることができないことがある。しかし、すでに詰め込まれていた物体を取り出すことによって詰められなかった物体を詰められる場合がある。

過去の状態のデータ・ベースをファイルに保存し、そのファイルを呼ぶことで、過去の状態まで戻ることを考える。本論文では物体を詰めるのに不適当な空間しか残っていない場合、すでに詰めてある二つの物体を取除いた状態までファイルをさかのぼり三つの物体の詰めこむ空間、順番を変えることによるシステムの向上を考える。第一に物体を詰める順番のすべての組合せを考える方法がある。しかしこの方法では組合せによる爆発が起こる可能性があり、また推論と呼ぶにはふさわしくない。そこでここではルール・ベースを利用した推論による方法を考える。推論方法は $\langle 4 \cdot 2 \rangle$ に基づいて行うが $\langle 4 \cdot 2 \rangle$ では 1 (物体) 対 m (空間)

図 20 各システムの詰め込み率

であったのに対し、ここでは物体の数が n 個 (この例では 3) になるため n (物体) 対 m (空間) での推論を行うことで評価する。また、物体をうまく詰められなかった原因として大きな物体が最後まで残ってしまった可能性も多いために、大きな物体から詰めるというルールを付け加えた。以上の推論より、バック・トラックに関して詰め替え可能なすべての物体のすべての空間に対する評価で最大のところに物体を詰めることができる。

7. システムの実行

実際に、A~H のルール群を図 18 に示すような組合せに基づいて、ルールの数増やした場合の物体を箱詰めしていったときの詰め込み率を算出した。データ (物体の寸法、詰め込む順番) は Rule Base6 で適当に詰められるように用いた。ルール数の増加のさせ方を図 19 に示す。空間選択のルール群を A,B,C,(Main Rule) とした、空間結合のルール群を D,E,F,G,(Sub Rule) とした。図 20 にルール群を増加させた場合の実行結果を示す。図中で■印は物体を箱詰めし終ったときに、残ってしまった空間の初期空間に対する割合を示している。●印は dead space の初期空間に対する割合を示している。

A : 空間の左右奥の物体、壁の有無による評価

B : 空間の左右奥の情報の幾何学的特徴による評価

C : 空間の左右奥の情報に接する情報の幾何学的特徴による評価

D : 隣合う空間の情報による結合

E : 隣合う空間に接する情報の幾何学的特徴による評価

F : 使用困難な空間の左右空間への結合

G : 使用不可能な空間の左右手前空間への結合

H : 詰め替えによるシステムの向上

図 20 により空間の選択に関するルール、結合に関するルールの増加に従い、all space, dead space の割

合が減少しているのがわかる。

価基準として示した

6. ま と め

文 献

本論文では基本的なルール・ベースの導入により、ある程度適当な空間に物体を積めることができることを示した。また基本的ルール・ベースに付加して、補足ルール・ベースの数を増やすことにより、広範囲の情報を得られ、より適した空間に物体を積められることができることを積み込みスペースと dead space を評

- (1) Winston, P.H.(白井・杉原訳), コンピューター・ビジョンの心理 (1979), 173-330, 産業図書.
- (2) 長尾, 知識と推論, (1989) 1-280, 岩波書店.
- (3) 安西, 認識と学習, (1989), 1-291, 岩波書店.
- (4) Clackson, W.F. and Mellish, C.S.(中村訳), Prolog フログラミング, (1983), 1, マイクロソフトウェア.