社団法人 情報処理学会　研究報告
IPSJ SIG Technical Report

2004－GI－12　(5)
2004／6／18

倉庫番における部分マップの組合せに基づく手詰り判定手法

小田原　大[†]　金子知適[†]　川合　慧[†]

コンピュータパズルの倉庫番のソルバにおける課題は、手詰りの局面を効率良く検出することであり、特に袋小路の判定は難しい問題として知られている。既存の手法では判定に再帰的な探索を必要とするため、判定の効率が悪い。本稿では、袋小路を含めた「手詰り」状態の一般的な判定方法を提案し、それを探索に組合せることを提案する。具体的には問題を区域に分割し、ユニットと呼ばれる各区域の状態の組合せによって効率的に手詰りの状態を判定する。実際に倉庫番の問題に適用したところ、既存の手法では判定が難しい局面に対しても手詰りと判定することに成功した。

A Method for Detecting Deadlocks Based on a Combination of Sub Maps in Sokoban

Masaru Odawara ,[†] Tomoyuki Kaneko [†] and Satoru Kawai[†]

A new method of detecting deadlock states is proposed. We define *unit* as a part of a map, and decompose a map into units. Each unit has gateways. The keeper can enter the unit through some gateways depending on status of other areas.

We judge whether the entire state is a deadlock or not by using combination of local status of units.

As a result, we could express a lot of deadlocks and showed ways of applying it to effective search.

1. はじめに

コンピュータパズル「倉庫番」は Pushing Block 系のパズルの一種とみなせ、その複雑性は PSPACE 完全 かつ NP 困難であることが知られている[3)2)]。ソルバは人間の解くことのできる全ての問題を解くことはできず、より強力なソルバの開発はゲーム研究における一つの課題であるといえる。

ソルバの開発において、袋小路[1)] と呼ばれる手詰りの状態の発見が困難とされている。

本研究では、このような手詰りの状態を効率的に発見することを目標として、倉庫番マップを部分に分割し、その関係に着目した。

部分マップの組合せを用いることにより、多様な場合を表現することができるため、部分マップのデータベースを形成すれば空間的にも効率的な判定が可能になると考えられる。

† 東京大学大学院総合文化研究科広域システム科学系
Department of General Systems Studies, Graduate School of Arts and Sciences, University of Tokyo
E-mail:{haradai,kaneko,kawai}@graco.c.u-tokyo.ac.jp

図 1　地形: 左から順に番人、荷物、ゴール、スペース、壁、ゴール上の荷物
Fig. 1　Elements of a map.

1.1 倉庫番とは

番人が、**壁**と**スペース**からなる**マップ**上に散らばる**荷物**を全て**ゴール**上に押して運び、配置することが目的のコンピュータ上で遊ぶ一人ゲームである。図 1 にマップ要素の凡例を、図 2 に問題例を示す。

1.2 関連研究

上野らのソルバの研究では、荷物と壁からなる「閉領域」を、回復不能な状態である「袋小路」と回復可能な「ほら穴」とにわけ、袋小路の判定が困難であることを指摘した[1)]。この判定手法では、一部の荷物を除いて閉領域回避探索を行なうが、この過程において別の閉領域が出現する可能性があるために、判定プロセスの再帰的な呼び出しが必要となる。

Andreas らはそのソルバの開発において、不能状態の発見のためにデータベースを用いた[4)]。

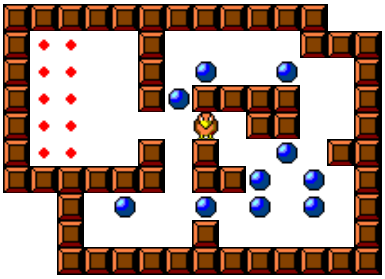

図 2　倉庫番の問題例
Fig. 2　An instance of Sokoban puzzle.

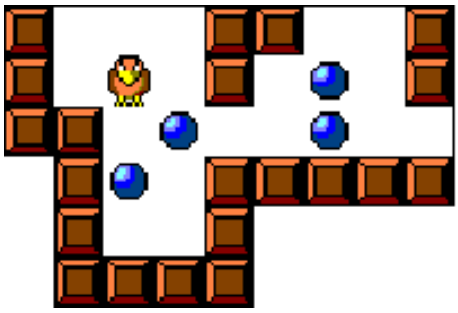

図 4　不能状態 の例
Fig. 4　An example of an infeasible state.

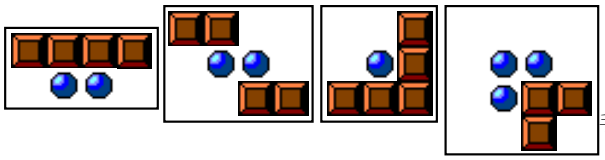

図 3　死に手の例
Fig. 3　Examples of the blocked state.

本稿は以下、2 章で「手詰り」を分類し、3 章で 2 種類の部分マップを定義し、それによるマップの表現の手法について触れる。4 章で手詰りの判定手法を説明し、5 章で具体的なマップに対する適用例を示し、先行研究で解決が困難な場合について手法を適用し、手詰りであることを示した。6 章にまとめと今後の課題を記す。

2. 倉庫番「手詰り」の分類

倉庫番では荷物を一度に一つだけしか押すことができず、また引くこともできない。そのため、誤った手順で荷物を押していくと荷物をゴールに運びこんでいないのにも関わらず、全く動かすことのできない状態が生じる場合がある。このような状態に陥ると、プレイヤーは手順を遡ってやり直すしかない。ここでは、ゴールに到達不可能な状態すべてを**手詰り**という。

この手詰りには 2 通りの場合がある。まず、全く動かせない荷物が存在するような状態である。図 3 がそのような状態の一例である。このような状態をここでは**死に手**という。

また、荷物を動かすことは可能だがそのまま進めていっても死に手の状態に必ず導かれるような状態をここでは **不能状態** であるという。図 4 がその一例である。

死に手の判定は部分的に行なうことができ、ほぼ自明であるといえるが、不能状態は周囲の状態との関連から生じるため、その判定は容易ではない。

3. マップのモデル化

本章ではマップを部分マップの組合せとみなす表現手法を提案する。

3.1 部分マップの組合せによる表現

マップの区域である部分マップを次の 2 つに分類する。

(1)　**ユニット**
(2)　**ユニット間通路**

ユニットとユニット間通路は、その出入口を繋ぎ合わせて互いに結ばれる。ユニットの間には必ずユニット間通路が存在するとする。部分マップのもつ出入口は他の部分マップの出入口とつながっているか、もしくは壁によって塞がれている。壁によって塞がれている場合には、その出入口は予め使えないとして無視される。

マップを部分マップに切り分けてその組合せとして表現することを**ユニット分割**と呼ぶ。

番人の位置

基本的に番人はユニットの内部にはいないものとする。ユニット分割の際に、ユニット内部に位置していた番人は、ユニットの出入口から荷物を動かさずに出ることのできる出入口を用いてユニット間通路に移動するものとする。

3.2 部分マップの定義と性質

まず、**ユニット間通路**を以下のように定義する。

定義　外に通じる通路 (**出入口**) を持ち、荷物を 1 つも含まない倉庫番マップの一部。

ユニット間通路にはゴールを含むことがある。また、面積を持たない場合も存在する☆。以下で触れる**到達可能性**を備える。

ユニットとは以下のようなものである。

定義　出入口と荷物をそれぞれ 1 つ以上含んでおり、

☆ 実際のマップ上でユニットが隣接している場合

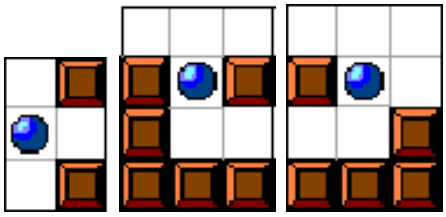

図 5 ユニットの例
Fig. 5 Instances of the *units*.

ゴールを含まない倉庫番マップの一部。
例えば図 5 の部分マップはユニットとみなすことができる。この場合、外に繋がりうる出入口の数は図の左からそれぞれ 6 つ、6 つ、7 つ存在する。

ユニットは以下の 2 つの性質を持つ。

(1) **通過可能性** 出入口を要素としてもつ順序対 (g_i, g_j)(**パス**と呼ぶ) の集合をいう。パスは以下の [i],[ii],[iii] のいずれかに順に分類される。

可能パス 番人が g_i から入ったとき、ユニット内部で手詰りを作りださずに g_j から出ることができる。ただし、そのユニットの内部の状態を変化させる場合がある。

- 内部の荷物の位置を変えるパス …[i]
- 内部の荷物の位置を変えないパス …[ii]

不可能パス g_i から入り、g_j から出ることができない、あるいはその際にユニット内部に手詰りが生じるようなパス …[iii]

(2) **出入口の利用可能性** ユニットの外に番人がいるとき、番人がどの出入口を利用かという性質。番人の位置によって変わる。探索によって変わる可能性がある。

内部の荷物の位置が変わると、同じユニットでも通過可能性が変化することが多い。したがって、そのパスを通過した場合、ユニット内部の通過可能性が変わる。

集合 G_{from} と G_{to} について、$G_{from} \rightarrow G_{to}$ と書くと、以下の集合 $path$ の要素が 「荷物の位置を変えない可能パスである」ことを意味する。また、$G_{from} \rightarrow^* G_{to}$ と書くと、「荷物の位置を変える可能パスである」ことを表す。$G_{from} \not\rightarrow G_{to}$ では、$path$ は不可能パスの集合となる。

$$path = G_{from} \times G_{to} = \{(g_i, g_j) | g_i \in G_{from}, g_j \in G_{to})\}$$

なお、ユニット U_i の n 個の出入口の集合 G_i は、$G_i = \{g_0^i, g_1^i, ..., g_{n-1}^i\}$ と表す。

荷物の押し出し

荷物を押しださなければユニットの出入口から出られない場合が存在する。この時には荷物を押しだしつつ通過可能であるとする。

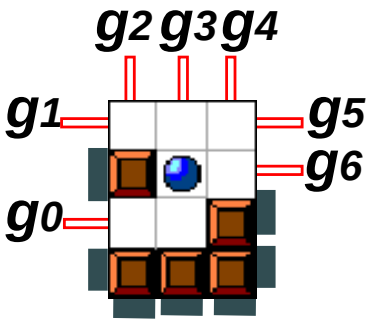

図 6 ユニット
Fig. 6 An unit.

3.3 例

図 5 の右図を、図 6 として表現できる。周囲の長方形は出入口を表す。濃いグレーは出入口でないところを表す。

このユニットの通過可能性は以下の通り表せる。

可能 (荷物の位置を変える)[i]

$$\{g_0\} \rightarrow^* \{g_1, g_2, g_3, g_4, g_5, g_6\} \quad (1)$$

可能 (荷物の位置を変えない)[ii]

(1), (3) 以外 (2)

不可能 [iii]

$$\{g_1, g_2, g_3, g_4, g_5, g_6\} \not\rightarrow \{g_0\} \quad (3)$$

(4)

(3),(4) については、パスが存在するが、このパスを用いた場合ユニットの通過可能性が変化する。パスの集合のサイズは $|G|^2$ であり、(1), (3) 以外のパスの集合についてはユニットの状態を変化させずに通過が可能なパスとみなす。

前述したように、これらの出入口は利用可能なものと不可能なものに分けられる。不可能な出入口を左に含むパスについてはその時点においては実質考慮の必要がない。

3.4 出入口の利用可能性

ユニットの出入口の利用可能性を求める。番人の到達可能な範囲を調べる。番人が移動可能な範囲を**可動範囲**といい、そうでない範囲を**非可動範囲** という。可動範囲は以下のように定める。

- ある時点において番人が荷物を一つも押すことなく移動できる範囲。…[1]
- 可動範囲に接する出入口 g_k との対が通過可能であるような出入口 g_l に接する範囲。…[2]

[1][2] のいずれかを満たす範囲は可動範囲であるとす

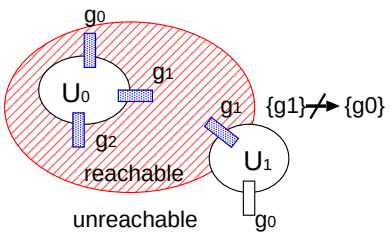

図 7 利用可能な出入口と不可能な出入口
Fig. 7 Some gates are available, the other is not.

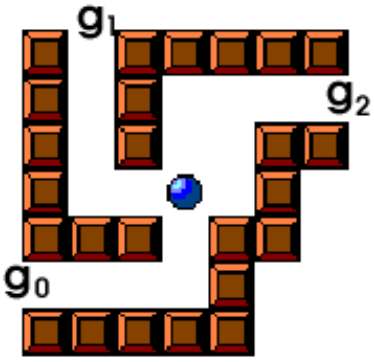

図 8 通過可能性が変化する場合
Fig. 8 A case passableness changes

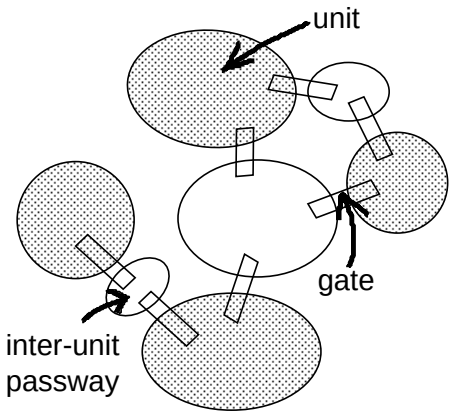

図 9 ユニットによるマップの表現
Fig. 9 An schematic diagram of a map by using units.

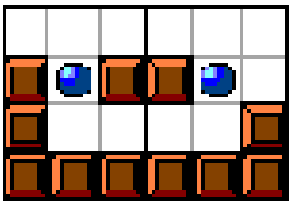

図 10 手詰りとなる組合せ
Fig. 10 A combination that results in a deadlock state.

る。あるユニット間通路が、可動範囲に含まれる時、そのユニット間通路は到達可能となる。

ユニットの出入口は、上で定めた可動範囲に接している時に利用可能である。これによってユニット出入口の利用可能性が定められる。

図 7 において、可動範囲が斜線部であるとしたら、図中のユニット U_0 の全ての出入口 $g_i^0 \in G_0$(図中グレーの出入口) を利用できる。一方、ユニット U_1 については可動範囲に接していない出入口 g_0^1 は利用できない。

3.5 通過可能性の変化

ユニットの入口の利用可能性を調べる過程で、他のユニットの通過可能性が変化する場合がある。例えば、図 8 においては

$$\{g_2\} \not\rightarrow \{g_0, g_1\} \tag{5}$$

である。しかし、パス (g_0, g_1) を通過すると (6) となり、これらのパスがともに通過可能となる。

$$\{g_2\} \rightarrow \{g_0, g_1\} \tag{6}$$

このとき、例えば g_0 から g_2 に移動し、状態を変化させた上で g_2 から g_1 に移動することができる。

本稿では、部分マップの関連性を図 9 のように表現するユニットはグレーの楕円で表し、その出入口はユニット内外を結ぶ長方形によって表す。白抜きの楕円はユニット間通路を表す。この模式的な表現をここでは**ユニット図**と呼ぶ。

4. 手詰りの判定

通過可能性と出入口をもったユニットがユニット間通路によって互いに結びつけられ配置されることを示した。ここで、**手詰り**は以下のように定義される。

定義 非可動範囲に接する出入口を持つユニットが存在するとき、その状態は**手詰り**である。

例

簡単な例を挙げて説明する。図 5 の中図のユニット (U_0 とする) と右図のユニット (U_1 とする) について調べる。明らかな手詰りの例として、これらの 2 つのユニットを中図を左に、右図を右に隣接させた配置について考える。番人はユニット間通路 P_1 にいるとする。

この場合の各ユニットの通過可能性と、入口の利用可能性について調べる。

U_0(図 11) は以下の通過可能性を持っている。

可能 (荷物の位置を変える)

$$\{g_5\} \rightarrow^* \{g_0, g_1, g_2, g_3, g_4\} \tag{7}$$

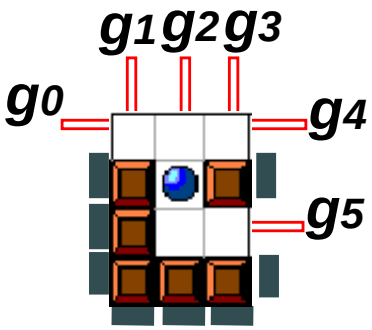

図 11 ユニット U_0
Fig. 11 An unit U_0.

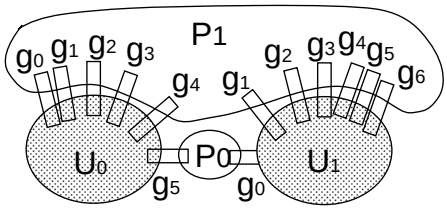

図 12 図 10 のユニット図
Fig. 12 A unit expression of Fig.10.

可能 (荷物の位置を変えない)

(7), (9) 以外 (8)

不 可 能

$$\{g_0, g_1, g_2, g_3, g_4\} \not\rightarrow \{g_5\} \quad (9)$$

U_1(図 6) の通過可能性は、前述した通り (1),(2),(3) によって示される。

これらより、2 つのユニットの結び付きを図で表現したものが図 12 となる。P は、U_0 と U_1 を結ぶユニット間通路である。

番人はユニット U_0, U_1 の外 P_1 に存在するから、利用可能な入口は P_1 に接する $g_0^0, g_1^0, g_2^0, g_3^0, g_4^0, g_1^1, g_2^1, g_3^1, g_4^1, g_5^1, g_6^1$ である。

P_0 に結び付いているユニットは U_0, U_1 の 2 つだけであるから、P_0 にはこれらの 2 つを通過しなければ到達できない。ここで、上の通過可能性 (9) より、番人は U_0 を通過して P_0 に到達することはできない。また、(3) より、U_1 を通過しても同様に P_0 に到達することはできない。

したがって、P_0 は非可動範囲となり、U_0、U_1 の 2 つがこれに接しているため、この状況は手詰りであると判定される。

2 つのユニットが通路を挟む場合 (図 13) も、これをユニット間通路とみなすことで同様に図 12 で表す

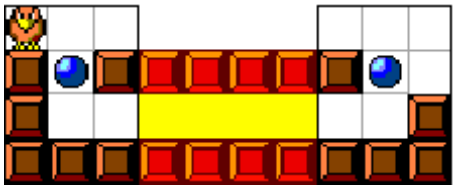

図 13 長さ 4 マスの通路を挟み込んだ場合
Fig. 13 A deadlock state that consists of two units and a passway.

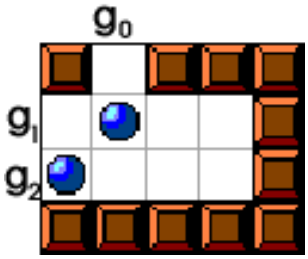

図 14 ユニット内部の探索
Fig. 14 Searching the inside of a unit.

ことができる。そのため手詰りと判定される。

5. 適 用

本章では実際の探索における適用の手法について触れる。

初期状態から番人の操作を繰り返して得られる局面を対象とし、まずこのマップをユニット分割する。

この際に各部分マップの性質については全て個別に求め、ユニット間の関連性を調べる際にはユニット内部についての探索は行なわない。

ユニット内部の探索

ユニットの通過可能性については、今まで定義されているものとして用いてきたが、実際の適用においてはその場で内部を探索して求める。例えば、図 14 では、パス $(g_0, g_1), (g_0, g_2)$ については番人を g_0 から g_1, g_2 へ移動させるということを実際に行なう。その過程で、ユニット内部でただちに死に手 (図 3) に導くため、不可能と判定される。

(g_1, g_0) は通過可能性を変化させる可能パスであり、(g_2, g_2) は 不能状態 (図 4) が発生するため不可能パスとされる。

探索の過程で部分的な手詰りがユニット内部で生じる場合、それは判定できることを前提とする。

番人の位置

ユニットに切り分けた際、番人がユニット内部に存在することになるが、この際番人の位置としては、ユニット内部ではなく、その場から荷物をひとつも動か

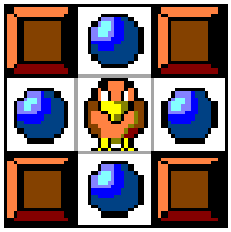

図 15 特別なユニットの例
Fig. 15 A unique case of a unit.

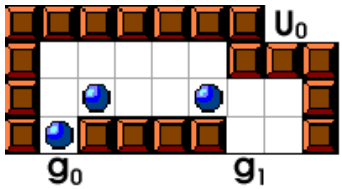

図 16 screen 2 のユニット U_0
Fig. 16 U_0 of screen 2.

さずに移動できる出入口に接するユニット間通路に移動することにする。

図 15 のようにユニットを切り出した場合、番人はユニット内部にしか存在することができないことになるが、探索過程でこのような状態に陥るのは初期状態における一局面だけである。普通の問題であれば初期状態は手詰りではありえないため、このような場合は対象としない。

マップの変換から判定までのプロセスは以下の順序で行なわれる。

(1) 手詰りかどうかを判定したい状態のマップを用意する。
(2) マップをユニットとユニット間通路に分割する。
(3) 現在の番人の位置を元に、ユニットの入口の利用可能性について調べる。
(4) 4 章で示した判定法を元に手詰りかどうかを判定する。

5.1 適 用 例

ユニットの組合せに基づく手詰り判定の例を示すために具体的なマップを元に適用してみる。例としては図 2 で示した Xsokoban[6] の 2 面の初期状態を用いる。

まず、ユニットを図 16, 図 17, 図 18 の 3 つに分ける。この時、倉庫番マップはユニット図により 図 19 のように示される。

ここで各ユニットの通過可能性は以下のように表される。

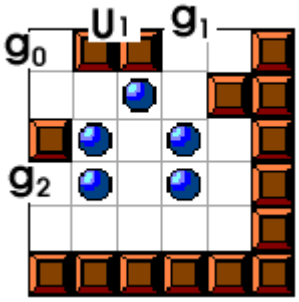

図 17 screen 2 のユニット U_1
Fig. 17 U_1 of screen 2.

図 18 screen 2 のユニット U_2
Fig. 18 U_2 of screen 2.

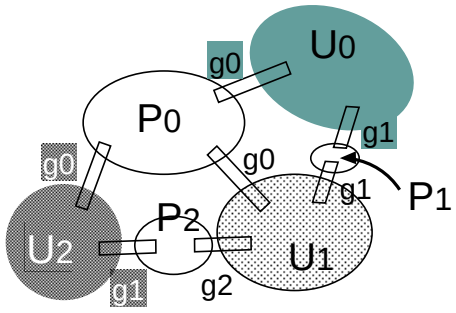

図 19 xsokoban screen 2 のユニット図
Fig. 19 An unit expression of xsokoban screen 2

$$U_0: \quad \{g_0^0\} \not\to \quad \{g_1^0\} \tag{10}$$
$$\{g_1^0\} \to^* \quad \{g_0^0\} \tag{11}$$
$$U_1: \{g_0^1, g_1^1\} \to^* \{g_2^1\} \tag{12}$$
$$\{g_0^1, g_2^1\} \to^* \{g_1^1\} \tag{13}$$
$$\{g_1^1, g_2^1\} \not\to \{g_0^1\} \tag{14}$$
$$U_2: \quad \{g_0^2\} \to^* \quad \{g_1^2\} \tag{15}$$
$$\{g_1^2\} \not\to \quad \{g_0^2\} \tag{16}$$

ここで、番人は図 19 の P_0 の領域にいるものとする。したがって P_0 は可動範囲となる。ユニット U_0 と U_1 の間、もしくは U_1 と U_2 の間が非可動範囲とされればこの状態は手詰りと判定される。

U_0-U_1 間

(10) より U_0 を通過して P_1 に到達することはでき

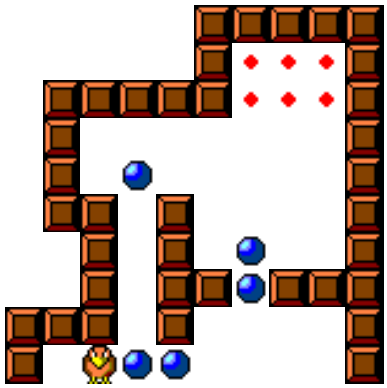

図 20 文献[1] において誤って判定される手詰り状態
Fig. 20 A deadlock judged incorrectly in the program[1].

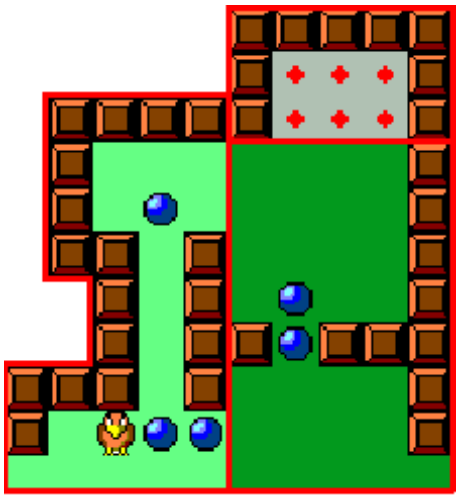

図 21 図 20 のマップの部分マップ分割
Fig. 21 Unit division of a map Fig.20.

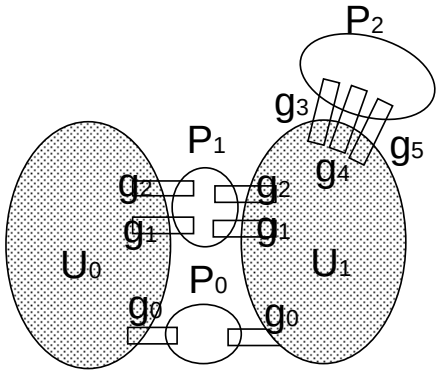

図 22 図 20 のマップのユニット図
Fig. 22 Schematic diagram of a map Fig.20.

ないが、(13) より U_1 を通過して P_1 に到達可能である。したがって P_1 は可動範囲となる。

U_1-U_2 **間**

(13) より U_1 を通過して P_2 に到達可能である。したがって P_2 は可動範囲となる。

これより、全てのユニット間通路が可動範囲と判定されるため、この状態 (図 2) は手詰りではない。

5.2 先行研究との比較

deadlock table による判定

Junghanns[4] は、5×4 の大きさの deadlock table を構成し、それによって手詰りの判定を行なった。しかしこの手法だとデータの大きさにはどうしても限界が生じるため、サイズが僅かに異なるだけでも判定が不可能になる。一方本研究においては、サイズの多少の変化に因われることなく手詰りを判定できる。

探索による判定

上野ら[1] は、閉領域探索のプロセスを呼び出すことで、新たに閉領域が生成される場合が生じた。このため再帰的にこのプロセスを呼び出す必要が生じた。また、以下の図においては一部の荷物を除く前処理のために、手詰りを手詰りでないと判定してしまう。

本研究の手法においては、このマップは図 20 のようにユニット分割することでただちに判定できる。

これをユニット図で表すと図 22 となる (一部、無関係な出入口については省略)。

現在番人はユニット間通路 P_0 にいるとみなすことができ、このとき $\{g_0^1\} \not\rightarrow \{g_1^1, g_2^1, g_3^1, g_4^1, g_5^1\}$ かつ $\{g_0^0\} \not\rightarrow \{g_1^0, g_2^0\}$ であり、P_1、P_2 が非可動範囲となる。したがってこの状態は手詰りであると判定できる。

5.3 判定の健全性

倉庫番の手詰りの判定に関して、完全 (手詰りは必ず手詰りと判定する) かつ健全 (手詰りと判定したら必ず手詰りである) であるような手法の構成は本質的な困難を抱えている。

したがって本研究では、健全性を損なわないことを目標とした。前述の荷物を押し出すという条件についてもそのためである。

実際に探索に用いる場合、完全性は保ち、健全性を損ねた場合には、解ける状態を「手詰り」と判定して探索を打ち切ってしまうが、一方本研究の手法の場合健全性が保たれるためその事態は避けられ、有効であるといえる。

6. おわりに

6.1 まとめ

本研究では、倉庫番のマップを状態を持つ部分マッ

プに切り分け、それらの関連によって手詰りを判定する健全な手法を提案した。この手法を用いた結果として、既存の手法では判定不能であるような場合についても手詰りを判定することができた。

6.2 今後の課題

マップをどのようにユニットに区切るかによって、手詰りを正しく判定する割合が大幅に変わり得る。実際に探索に用いる際にはできる限り効率的に、部分マップに切り分ける手法を検討する必要がある。

また、今回の手法ではユニット中にゴールを配置することを考慮しなかった。ゴールが連続して隣接して一部に集中している配置のマップならこのような手法で実用的であると思われるが、ゴールが散在するような配置に対しても今後は考慮する必要があるだろう。

謝　　辞

アドバイスを頂いた GPS の皆様に感謝します。

参 考 文 献

1) 上野篤、中山康、疋田輝雄 第 3 章 倉庫番 松原仁・竹内郁雄編 ゲームプログラミング, 共立出版, (1998), pp.158-172
2) Dor, D., Zwick, U. Sokoban and other motion planning problems. *Comput. Geom. Theory Appl. 13, 4* (1999), pp.215-228
3) Joseph C. Culberson, Sokoban is PSPACE-complete, Technical Report TR 97-02 April 1997, DEPARTMENT OF COMPUTING SCIENCE The University of Alberta.
4) Andreas Junghanns, Pushing the Limits: New Developments in Single-Agent Search, Department of Computing Science University of Alberta Ph.D.thesis, 1999.
5) “Rolling Stone”, http://www.cs.ualberta.ca/%7Egames/Sokoban/program.html
6) Xsokoban, http://www.cs.cornell.edu/andru/xsokoban.html
7) Adi Botea, Martin Müller, Jonathan Schaeffer, Using Abstraction for Planning in Sokoban, CG 2002, LNCS 2883, pp. 360-375, 2003