情報処理学会論文誌　プログラミング　Vol.5　No.3　50–60 (Aug. 2012)

多重Ambient Calculusによる物流記述に対する弱双模倣等価性を用いたモデル検査

樋口 昌宏[1,a)]　森田 哲平[1]　加藤 暢[1]

受付日 2011年12月16日, 採録日 2012年3月21日

概要：我々は Ambient Calculus（AC）による物流システムの記述とそのモデル検査に関する研究を進めている．モデル検査において数千個規模の貨物輸送を扱おうとすると，プロセス式が巨大かつ複雑なものとなるため，状態空間爆発がおこり検証が実際上不可能になるという問題がある．これに対処するため，我々は個々の貨物の取扱いを個別のプロセス式として記述することが可能な多重 Ambient Calculus（MAC）を提案し，プロセス式間の弱双模倣等価関係を導入し，その性質について議論した．本論文では，MAC により記述された物流システムのモデル検査法を提案し，それに基づくモデル検査システムについて述べる．提案する検査法では，多数の貨物に対する輸送計画を記述した式 P が，ある貨物 c の輸送に関する所期の性質 $f(c)$ を満たしていることの検証を，貨物 c に対する輸送計画を含むより小規模な式 Q を用いて以下の 2 つの性質の検証に帰着させる．(i) P と Q が貨物 c の輸送について弱双模倣等価である（振舞いが等しい）こと，(ii) Q が $f(c)$ を満たすこと．本論文では，この手法に基づくモデル検査システムの構築，ならびに検証実験についても述べる．

キーワード：プロセス代数，移動型プロセス，検証

Model Checking for Freight Systems Written in the Multiple Ambient Calculus Using Weak Bisimulation

Masahiro Higuchi[1,a)]　Teppei Morita[1]　Toru Kato[1]

Received: December 16, 2011, Accepted: March 21, 2012

Abstract: We are investigating the way for describing freight systems in the Ambient Calculus and constructing freight management systems based on it. We noticed the state space explosion problem that prevented us from model checking when we treated the processes representing the transporting plans for thousands of containers, even though we introduced partial order reduction methods. In order to solve the problem, we proposed the Multiple Ambient Calculus(MAC), which enables us to model freight systems by a set of formulas, introduced the weak bisimulation relation between the processes of MAC, and showed several properties of the relation. This paper proposes a verification method for freight systems written in MAC and discusses a model checking system using the method. Let P be the expression representing the transporting plan for large number of containers, $f(c)$ be the formula representing the desirable property for a container c, and Q be the expression representing the transporting plan for the container c. The verification if P satisfies $f(c)$ is reduced to the following two properties: (i)P is weak bisimilar to Q on the transportation of c, (ii)Q satisfies $f(c)$. This paper also describes the construction of the model checking system based on the method and the results of several experiments using the system.

Keywords: process algebra, process mobility, verification

1. はじめに

近年，物流の世界では，貨物の流通量の増加にともない，コンテナ管理の重要性が高まっており，コンテナの管理方

[1] 近畿大学
Kinki University, higashiosaka, Osaka 577–8502, Japan
[a)] higuchi@info.kindai.ac.jp

法についてさまざまな研究が行われている [1], [2], [3]．また，物流の大規模化にともない，物流管理システムの信頼性の重要性が大きくなってきている．そのような物流管理システムを構築する基盤として，物流計画を記述するための形式的な枠組みが必要になってくると考えられる．その枠組みを確立すれば，物流計画の厳密な定義が行え，物流計画そのものが所期の性質を満たしていることを形式的に検証することが可能となる．

我々は，すでにそのような形式的枠組みとして Ambient Calculus [4] を利用し，Ambient Calculus による物流計画の記述と，記述に基づく物流監視システム [5]，モデル検査システム [6] を開発している．モデル検査システムでは多数の貨物を扱うことによるモデル検査の際の状態空間爆発を緩和するため，プロセス式の対称性の検出と，対称性を利用した partial order reduction を行っている．しかし，そのような方法を用いたとしても検査可能な物流計画はコンテナ数 200 個程度が限界であり，現実の物流計画ではコンテナ数が数千個規模であることを考えると，さらなる効率化が必要と考えられる．

一方，我々は Ambient Calculus を用いた物流計画記述の記述性，可読性の向上を目的とした多重 Ambient Calculus を提案し，プロセス式の間の弱双模倣等価関係を定義した [7]．多重 Ambient Calculus を用いることにより物流計画の持つ対称性，並行性を陽に表現した記述が可能になる．本論文では多重 Ambient Calculus により陽に表現された物流計画の対称性を体系的に利用し，大規模なプロセス式に対するモデル検査をその式と弱双模倣等価なより小規模なプロセス式のモデル検査に帰着させる手法について述べる．また，提案手法に基づき拡張したモデル検査システムとそれを用いた検証実験についても述べる．

2. 多重 Ambient Calculus による物流記述

2.1 移動プリミティブを持つ Ambient Calculus

文献 [4] において，通信プリミティブを含む Ambient Calculus のサブクラスとして，移動プリミティブのみを備えた Ambient Calculus が定義されており，チューリング機械を模倣できるという意味で万能であることが示されている．本論文では，物流システムのモデル化のために移動プリミティブのみを備えた Ambient Calculus を利用することを考える．以下にその構文規則およびプロセス式間の基本的な合同関係（構造合同）の定義を示す．

定義 2.1（構文規則）

n	names		$M ::=$	capabilities
$P, Q ::=$	processes		$in\ n$	can enter n
$(\nu n)P$	restriction		$out\ n$	can exit n
0	inactivity		$open\ n$	can open n
$P \mid Q$	composition			
$!P$	replication			
$n[P]$	ambient			
$M.P$	action			□

通常，プロセス式中の「(」,「)」および「0」は混乱のない範囲で省略される．

定義 2.2（構造合同）

$P \equiv P$	(Refl)
$P \equiv Q \Rightarrow Q \equiv P$	(Symm)
$P \equiv Q,\ Q \equiv R \Rightarrow P \equiv R$	(Trans)
$P \equiv Q \Rightarrow (\nu n)P \equiv (\nu n)Q$	(Res)
$P \equiv Q \Rightarrow P \mid R \equiv Q \mid R$	(Par)
$P \equiv Q \Rightarrow\ !P \equiv !Q$	(Repl)
$P \equiv Q \Rightarrow n[P] \equiv n[Q]$	(Amb)
$P \equiv Q \Rightarrow M.P \equiv M.Q$	(Action)
$P \mid Q \equiv Q \mid P$	(Par Comm)
$(P \mid Q) \mid R \equiv P \mid (Q \mid R)$	(Par Assoc)
$!P \equiv P \mid !P$	(Repl Par)
$(\nu n)(\nu m)P \equiv (\nu m)(\nu n)P$	(Res Res)
$(\nu n)(P \mid Q) \equiv P \mid (\nu n)Q$ if n が P の自由変数でない	(Res Par)
$(\nu n)(m[P]) \equiv m[(\nu n)P]$ if $n \neq m$	(Res Amb)
$P \mid 0 \equiv P$	(Zero Par)
$(\nu n)0 \equiv 0$	(Zero Res)
$!0 \equiv 0$	(Zero Repl)
	□

定義 2.2 および以降では「⇒」は含意を表す．

以降では互いに構造合同なプロセス式は同一視することとし，特に (Repl Par) より「$!P \equiv !P \mid \underbrace{P \mid \ldots \mid P}_{n}\ (1 \leq n)$」が得られるが,「$!P$」をそれらの標準形と考え,「$!P \mid P \mid \ldots \mid P$」のような式は考慮の対象としない．

Ambient Calculus では，ambient 構文「$n[P]$」を用いて内部にプロセスを持つオブジェクト（以降アンビアントと呼ぶ）を表現することができ，これを再帰的に用いることによりアンビアント間の入れ子構造を表現できる．たとえば $l[m[n[0]]]$ によりアンビアント l の中にアンビアント m が存在し，さらにその中にアンビアント n が存在するというような階層構造を表現できる．以降ではアンビアント l の中にアンビアント m が存在するとき，l が m の親，m が l の子であるという．親子関係の反射推移的閉包を祖先子孫関係という．

ambient 構文と action 構文を用いて「$n[M.P]$」のような形でアンビアント内に記述された capability（ケーパビリティ）M により，階層構造の動的な変動を記述できる．ケーパビリティを消費することにより，プロセス式は異な

る階層構造を持つ式に遷移する．式の遷移規則を以下に示す．

定義 2.3（遷移規則）

$n[in\ m.P \mid Q] \mid m[R] \to m[n[P \mid Q] \mid R]$ (In)
$m[n[out\ m.P \mid Q] \mid R] \to n[P \mid Q] \mid m[R]$ (Out)
$open\ n.P \mid n[Q] \to P \mid Q$ (Open)
$P \to Q \Rightarrow (\nu n)P \to (\nu n)Q$ (Res)
$P \to Q \Rightarrow n[P] \to n[Q]$ (Amb)
$P \to Q \Rightarrow P \mid R \to Q \mid R$ (Par)
$P' \equiv P, P \to Q, Q \equiv Q' \Rightarrow P' \to Q'$ (Cong) □

遷移規則（In）は，n アンビアントが自身の持つケーパビリティ「$in\ m$」を消費し，m アンビアントの中に入る遷移を表す．遷移規則（Out）は，n アンビアントがケーパビリティ「$out\ m$」を消費し，m アンビアントの外へ出る遷移を表す．また，遷移規則（Open）は，ケーパビリティ「$open\ n$」が消費されることにより n アンビアントが消滅し，n が内部に保持していたプロセスはその親に継承される遷移を表している．「$\to$」の反射推移的閉包を「$\overset{*}{\to}$」で表す．

遷移規則の定義では ambient 構文「$n[P]$」を用いて記述されたアンビアントであっても，action 構文「$M.P$」を用いて何らかのケーパビリティに先行されているもの，およびその子孫のアンビアントは遷移に関与しない．そのようなアンビアントを非活性といい，そうでないものを活性ということにする．

Ambient Calculus では，「$Name \triangleq P$」というようなプロセス定義式によりプロセス式に名前を与えておいて，別の（自身でもよい）プロセス式中に $Name$ と書くことでプロセス式を呼び出すことができる．一般には再帰的なプロセス定義式も許される．

動的に変動するアンビアント間の階層構造を記述できるという Ambient Calculus の特徴を利用して，物流システムを表現できる．これを簡単な記述例を用いて説明する．

例 2.4　船 SHIP が東京港（TK）でコンテナヤード（CY）からコンテナ CT を積み込んで神戸港（KB）へ輸送する物流計画を，以下の式により表現することができる．

```
SHIP[ in TK.
          (load[out SHIP. in CY. in CT]
        | open lcomp. out TK. in KB)]
| TK[ CY[ CT[open load. out CY. in SHIP.
          lcomp[out CT] ] ] ]
| KB[ CY[] ]
```

この式からケーパビリティと，ケーパビリティに先行されるプロセス式をすべて削除して得られる式

```
SHIP[] | TK[ CY [ CT[] ] ] | KB[ CY[] ]
```

は，船（SHIP）アンビアント，東京港（TK）アンビアントと神戸港（KB）アンビアントが並行して存在し，TK と KB はそれぞれ内部にコンテナヤード（CY）アンビアントを持ち，TK 内の CY にはコンテナ（CT）アンビアントが 1 つ存在するという，物流システムの現在の状態を表していると考えることができる．また，上記の式は

```
TK[ SHIP[ load[out SHIP. in CY. in CT]
        | open lcomp. out TK. in KB]
  | CY[ CT[ open load. out CY. in SHIP.
          lcomp[out CT] ] ] ]
| KB[ CY[] ]
```

という式に遷移可能で，その遷移は船が東京港に入港したという動作を表現していると考えることができる．この遷移により船が入港したこと，すなわち積み込みが可能になったことをコンテナに知らせる役割を持つ load アンビアントが活性化され，load アンビアントの一連の遷移により，

```
TK[ SHIP[ open lcomp. out TK. in KB]
  | CY[ CT[ open load. out CY. in SHIP.
          lcomp[out CT]
          | load[] ] ] ]
| KB[ CY[] ]
```

という式に遷移する．ここで CT アンビアントは open load ケーパビリティを消費することにより，引き続く荷物の積み込みを表す遷移 out CY と in SHIP が可能となり，

```
TK[ SHIP[ open lcomp. out TK. in KB
        | CT[ lcomp[out CT] ] ]
  | CY[] ]
| KB[ CY[] ]
```

へ遷移する．さらにこの式では積み込み完了を船に伝える lcomp アンビアントが活性化される．lcomp アンビアントが CT アンビアントを出ることにより SHIP の open lcomp ケーパビリティを消費可能となり，引き続く東京港からの出港，神戸港への入港を表す遷移 out TK. in KB が行われ，

```
TK[ CY[] ]
| KB[ SHIP[ CT[] ] | CY[] ]
```

へ遷移する．□

このように必要な同期をとりながらモノを輸送する物流計画を，アンビアントの階層構造の遷移列として表現することができる．以降では上記式のコンテナ名，積荷港名，荷降ろし港名を変数にした以下のプロセス定義式を引用することにする．この式は通常の海上輸送に用いられる送り状 [8], [9]（invoice）の記載内容を表現したものである．

定義 2.5（Invoice 式）

```
Invoice(x, y, z) ≜
  SHIP[ in y. ( load[out SHIP.in CY. in x]
              | open lcomp. out y. in z.
                uload[in x]
```

```
                | open ulcomp). out z ]
| y[ CY[ x[open load. out CY. in SHIP.
          lcomp[out x]
        | open uload. out SHIP. in CY.
          ulcomp[out x. out CY. in SHIP]] ] ]
| z[ CY[] ]
```
□

同様に，以降の便宜のため，東京港（TK），神戸港（KB），門司港（MJ）の順に入港する船 SHIP の航路のみを表現した式を定義しておく．

定義 2.6（SHIProute 式）

```
SHIProute≙
  SHIP[ in TK. out TK. in KB. out KB. in MJ]
| TK[] | KB[] | MJ[]
```
□

2.2　物流システム記述のための Ambient Calculus

物流システム記述の便宜をはかるために，Ambient Calculus にいくつかの定義を追加するとともに，いくつかの制限を課す．

一般のプロセス式 P から活性アンビアント間の階層構造のみを表すプロセス式を抽出する関数 $\mathcal{H}$ を，以下のように定義する．

定義 2.7（アンビアント階層の抽出）

$\mathcal{H}((\nu n)P) = (\nu n)(\mathcal{H}(P))$	restriction
$\mathcal{H}(0) = 0$	inactivity
$\mathcal{H}(P \mid Q) = \mathcal{H}(P) \mid \mathcal{H}(Q)$	composition
$\mathcal{H}(!P) = 0$	replication
$\mathcal{H}(n[P]) = n[\mathcal{H}(P)]$	ambient
$\mathcal{H}(M.P) = 0$	action

□

$\mathcal{H}(P)$ は，プロセス式 P が表す物流システムの現在の状態を表現しているものと考えることができる．そのような $\mathcal{H}(P)$ に現れるアンビアントは船やコンテナなどの物理的なモノを表すものであるか，例 2.4 で用いた load や lcomp のような何らかの役割を持ったインスタンス化されたソフトウェアオブジェクトを表すもののいずれかである．どちらにしても，何らかの手順を経て実体化されたものなので，$\mathcal{H}(P)$ が無限となるような P は物流システムを表現しているとは考えられない．一方，通常の Ambient Calculus では再帰的プロセス定義を許しているため，$P \triangleq n[P]$ というようなプロセス定義式を書くことにより簡単に無限の階層構造を持つプロセスが定義できてしまい，これも不適当である．

本研究では複製や再帰により無限に展開可能なプロセス式は排除しないが，$\mathcal{H}(P)$ が無限になるようなプロセス式 P は議論の対象としない．そのため，プロセス式定義に以下の制約を付加することとする．

定義 2.8（プロセス定義式の制約）

プロセス定義式 $Name \triangleq P$ において，右辺の P が（相互再帰を含む）再帰的なプロセス呼び出し $Name'$ を含む場合，それらの再帰呼び出しは $M.Name'$ の形で何らかのケーパビリティに先行されなければならない．□

この制限をプロセス定義式の記述に課することにより，アンビアント階層の定義 $\mathcal{H}(M.P) = 0$ より，再帰呼び出しの部分はアンビアント階層を求める際には無視され，結果として任意のプロセス式 P に対して $\mathcal{H}(P)$ は有限となる．

便宜上プロセス定義式の展開は式の遷移に影響を与えるとき，すなわち，プロセス呼び出しが活性アンビアント中でケーパビリティに先行されていない場合のみ展開することとし，「$n[in\ m.Name]$」のような式では $Name$ はそのまま放置することにする．

例 2.4 では TK（東京港）と KB（神戸港）内に同じ名前の CY（コンテナヤード）アンビアントがあるが，これらは別のものを表していると考えるべきである．このため本論文では同じ名前のアンビアントが存在する場合，遷移規則でケーパビリティと照合されるアンビアント名とは別に，一意の ID をそれらのアンビアントに与えておくことで，それらを区別するものと考える．そのため，プロセス式 P 中に同じ名前のアンビアントが存在する場合，活性であるないにかかわらず，あらかじめ何らかの ID が与えられているものとし，表記上はアンビアント名の添字で記述することにする．一方，複製演算子やプロセス呼び出しで動的に生成されるアンビアントについては，複製が生成されたとき，もしくは展開されたときに ID が付与されるものと考え，やはり添字で表記する．

プロセス式の遷移によりアンビアント階層は変化する．アンビアント階層の変化を遷移ラベルで表現することにする．遷移規則ごとのラベリング規則を以下のように定める．

定義 2.9（遷移ラベル）

$$n[in\ m.P \mid Q] \mid m[R] \rightarrow_l m[n[P \mid Q] \mid R],\quad l = \text{“}n \text{ enter } m\text{”} \qquad \text{(In)}$$

$$m[n[out\ m.P \mid Q] \mid R] \rightarrow_l n[P \mid Q] \mid m[R],\quad l = \text{“}n \text{ exit } m\text{”} \qquad \text{(Out)}$$

$$open\ n.P \mid n[Q] \rightarrow_l P \mid Q,\ l = \text{“}n \text{ disappear”} \qquad \text{(Open)}$$

□

遷移ラベル中のアンビアント名は ID 付きのものとする．このため

$$!(n[in\ m \mid P]) \mid m[] \rightarrow !(n[in\ m \mid P]) \mid m[n[P]]$$

のような複製の生成をともなう遷移の際には，遷移前に複製が必要な数だけ生成され（上記の場合 $n[in\ m \mid P]$ が 1 個），それとともにそれらの複製中の活性化されるアンビアント（上記の例では n，および P を単独のプロセス式と見た場合の P 中での活性アンビアント）に ID が与えられ，遷移ラベル中の n は与えられた ID を含むものと考える．

$P_0 \rightarrow_{l_1} P_1, P_1 \rightarrow_{l_2} P_2, \ldots, P_{n-1} \rightarrow_{l_n} P_n$（$n \geq 0$）であ

るとき $P_0 \xrightarrow{*}_{l_1 l_2 \ldots l_n} P_n$ と書くことにする．

Ambient Calculus で物流システムのような状態遷移システムをモデル化する場合，遷移ラベルはそのシステムで生起したイベントを表現していると見なすことができる．以降では遷移ラベルとイベントを同一視することにする．

2.3 多重 Ambient Calculus

文献 [5] で示した Ambient Calculus による物流記述は，単一のプロセス式で物流計画全体を記述するものであった．これに対して，多重 Ambient Calculus では n 個のプロセス式の組 $\overline{P} = (P_1, \ldots, P_n)$ で 1 つの物流計画全体を表現することを考える．以下では個々のプロセス式 P_i を個別式と呼び，$\overline{P}$ を全体式と呼び，全体式で表す物流計画全体をプロセス系と呼ぶことにする．

2.3.1 大域アンビアント

多重 Ambient Calculus では，アンビアント名の名前空間を，個々の個別式のみに関連する個別アンビアントのためのものと，一般に複数の個別式に共通に現れる大域アンビアントのためのものに分割する．全体式 $\overline{P}$ の大域アンビアント名の集合を $A_G(\overline{P})$ と書き，$\overline{P}$ の第 i 成分で用いられるアンビアント名の集合を $A^i(\overline{P})$，大域アンビアント名の集合を $A^i_G(\overline{P})$，個別アンビアント名の集合を $A^i_I(\overline{P})$ と書く．以降では，大域アンビアント名は先頭に大文字を用いて区別することとする．また一般性を失うことなく，1 つの全体式中の異なる個別式に同じ名前の個別アンビアントは存在しない（$A^i_I(\overline{P}) \cap A^j_I(\overline{P}) = \emptyset$）ものとする．また文脈から $\overline{P}$ が明らかなときには，$A_G(\overline{P})$，$A^i(\overline{P})$，$A^i_G(\overline{P})$，$A^i_I(\overline{P})$ は単に A_G，A^i，A^i_G，A^i_I と表記するものとする．

個別式のアンビアント階層 S から，指定したアンビアント名の集合 X を持つもののみを取り出す射影関数 $\mathcal{G}_X$ を以下のように定義する．

定義 2.10（アンビアント名射影）

$\mathcal{G}_X((\nu n)P) = (\nu n)\mathcal{G}_X(P)$	restriction
$\mathcal{G}_X(0) = 0$	inactivity
$\mathcal{G}_X(P \mid Q) = \mathcal{G}_X(P) \mid \mathcal{G}_X(Q)$	composition
$\mathcal{G}_X(n[P]) = n[\mathcal{G}_X(P)]$, if $n \in X$	glob. ambient
$\mathcal{G}_X(n[P]) = \mathcal{G}_X(P)$, if $n \notin X$	indi. ambient

□

定義 2.11（大域アンビアント階層）

全体式 $\overline{P} = (P_1, \ldots, P_n)$ について，$\mathcal{G}_{A_G}(\mathcal{H}(P_i)) = \mathcal{G}_{A^i}(H)$ $(1 \le i \le n)$ なる大域アンビアントのアンビアント階層 H が存在するとき，これを $\overline{P}$ の大域アンビアント階層といい，その集合を $\mathcal{H}_G(\overline{P})$ で表記する．□

たとえば

```
(KB[SHIP[]] | MJ[],
 TK[] | MJ[] | SHIP[])
```

に対して，`KB[SHIP[]] | TK[] | MJ[]` はその式の唯一の大域アンビアント階層である．一方，

```
(KB[SHIP[]] | MJ[],
 TK[SHIP[]] | MJ[])
```

では $\mathcal{G}_{A_G}(\mathcal{H}(P_i)) = \mathcal{G}_{A^i}(H)$ を満たす H として `TK[KB[SHIP[]]] | MJ[]` もしくは `KB[TK[SHIP[]]] | MJ[]` が存在し，大域アンビアント階層は一意ではない．

以降，全体式 $\overline{P}$ に個別式 Q を追加した全体式を $(\overline{P}, Q)$ と書くことにする．

2.3.2 遷移規則

全体式 $\overline{P}$ に対して，遷移ラベルの集合を $\mathcal{L}(\overline{P})$ と書き，遷移ラベルに現れるアンビアントがともに（disappear の場合は 1 つ）$A^i(\overline{P})$ の要素であるような遷移ラベルの集合を $\mathcal{L}^i(\overline{P})$ と書く．また，それらがともに $A_G(\overline{P})$ の要素であるような遷移ラベルの集合を $\mathcal{L}_G(\overline{P})$ とし，そのようなラベルを持つ遷移を大域遷移という．また $\mathcal{L}^i_I(\overline{P}) = \mathcal{L}^i(\overline{P}) - \mathcal{L}_G(\overline{P})$ とし，そのようなラベルを持つ遷移を第 i 成分の個別遷移という．$\mathcal{L}(\overline{P})$，$\mathcal{L}_G(\overline{P})$，$\mathcal{L}^i(\overline{P})$，$\mathcal{L}^i_I(\overline{P})$ は文脈に応じて $\mathcal{L}$，$\mathcal{L}_G$，$\mathcal{L}^i$，$\mathcal{L}^i_I$ と略記する．

混乱のない範囲で，$l \notin \mathcal{L}^i$ のラベルを持つ遷移で第 i 成分 P_i が変化しない場合でも，$P_i \to_l P_i$ と書くこととする．

全体式の遷移規則を以下のように定める．

定義 2.12（全体式の遷移規則）

$$
\begin{aligned}
&P_i \to_l Q_i \wedge l \in \mathcal{L}^i_I \wedge \mathcal{H}_G(\overline{P}) \neq \emptyset \\
&\quad \wedge \mathcal{G}_{A_G}(\mathcal{H}(P_i)) = \mathcal{G}_{A_G}(\mathcal{H}(Q_i)) \\
&\Rightarrow \overline{P} = (P_1, \ldots, P_i, \ldots, P_n) \\
&\quad \to_l \overline{Q} = (P_1, \ldots, Q_i, \ldots, P_n) \qquad \text{(individual)} \\
&\forall i(\ P_i \to_l Q_i) \wedge l \in \mathcal{L}_G \wedge \mathcal{H}(\overline{P}) \neq \emptyset \wedge \mathcal{H}(\overline{Q}) \neq \emptyset \\
&\Rightarrow \overline{P} = (P_1, \ldots, P_n) \to_l \overline{Q} = (Q_1, \ldots, Q_n) \quad \text{(global)}
\end{aligned}
$$

□

たとえば

```
P = ((KB[SHIP[out KB]] | TK[]),
     (TK[] | MJ[] | SHIP[]))
Q = ((KB[] | TK[] | SHIP[]),
     (TK[] | MJ[] | SHIP[]))
```

に対して $\overline{P} \to_{\text{“}SHIP\ exit\ KB\text{”}} \overline{Q}$ である．

遷移条件に遷移元で大域アンビアント階層を持つことを条件としているので，大域アンビアント階層を持たない全体式からの遷移は存在しない．

多重 Ambient Calculus では，たとえば定義 2.5 の Invoice 式と，定義 2.6 の SHIProute 式を組み合わせた定義 2.13 の $\overline{P}$ のような全体式を考えることができる．以降ではそのような形の全体式を検査対象とする．

定義 2.13（物流計画を表すプロセス式）

$\overline{P} = (P_1, P_2, \ldots P_n)$

$P_i = SHIProute_i \quad (1 \le i \le N_0)$

$P_i = Invoice(c_i, S_k, D_k) \quad (N_{k-1} < i \le N_k)\ (1 \le k \le R)$

ただし

N_0：輸送に用いる船の数

R：積荷港，荷降ろし港で分類した貨物の種別の数
$N_k - N_{k-1}$：k 番目の種別の貨物の数

3. 物流システムのための Ambient Logic

本論文では，プロセス式が物流システムの所期の性質を満たしているか検証するために，様相論理の一種である Ambient Logic [10] を用いる．ただし，文献 [10] で定義されているすべての様相記号に対応した検証系の構築は容易ではないため，文献 [6] において，それらの中で物流システムに求められる性質を記述するために必要となる様相記号だけを持つ，限定的な Ambient Logic を導入した．

3.1 物流システムのための Ambient Logic

文献 [6] において，物流システムが所期の性質を満たしていることを表す定義 3.1 に示すような性質を定めた．

定義 3.1（対象とする性質）
p1 いつか必ず貨物（コンテナ）は目的地に輸送される．
p2 特定の場所以外での貨物（コンテナ）の積み下ろしは行われない．（不正な貨物の移動の禁止）
p3 コンテナ船には決められた貨物（コンテナ）が積み込まれる．　□

本論文では，時制に関係する性質は全体式に対してのみ検査するため，限定的な Ambient Logic を以下のように非時相論理式と時相論理式に分けて定義する．

定義 3.2（論理式）
（非時相論理式）

η	names	
$\mathcal{A}\ \mathcal{B}$	::=	
	T	*true*
	$\neg\mathcal{A}$	*negation*
	F	$false(=\neg T)$
	$\mathcal{A}\vee\mathcal{B}$	*disjunction*
	$\mathcal{A}\wedge\mathcal{B}$	*conjunction*
	$\mathcal{A}\Rightarrow\mathcal{B}$	$implication(=\neg\mathcal{A}\vee\mathcal{B})$
	$\mathcal{A}\mid\mathcal{B}$	*composition*
	$\eta[\mathcal{A}]$	*location*
	$\blacklozenge\mathcal{A}$	*somewhere modality* [*1]

（時相論理式）

$\mathcal{T}\ \mathcal{U}$	::=	
	$\mathcal{A}$	
	$\neg\mathcal{T}$	
	$\mathcal{T}\vee\mathcal{U}$	
	$\Diamond\mathcal{T}$	*sometime modality*
	$\Box\mathcal{T}$	*everytime modality*

□

定義 3.2 において，*composition, location, somewhere modality* は，文献 [10] で導入されている Ambient Logic 独自の様相記号の一部である．定義 3.2 のこれら以外の論理記号は通常の時相論理で用いられる記号である．

定義 3.3（極大非時相式）　論理式 f において，定義 3.2 の時相論理式に現れる $\mathcal{A}$ の形をした極大な部分式，つまり $\Diamond$ や $\Box$ を含まないような f の極大な部分式を，f の極大非時相式と呼ぶ．　□

定義 3.2 によって規定される論理の意味は，定義 3.5 と定義 3.6 により与えられる．そのために必要となる記号を，定義 3.4 にまとめる．

定義 3.4

Π	個別式全体からなる集合
$\overline{\Pi}$	全体式全体からなる集合
Φ	論理式全体からなる集合
Φ^{-t}	非時相論理式全体からなる集合
$\mathcal{N}$	名前全体からなる集合
$P\downarrow P'$	$\stackrel{\triangle}{=}\exists n, P''.P\equiv n[P']\mid P''$
$\downarrow^*$	$\downarrow$ の反射推移的閉包
$\mathcal{R}(\overline{P})$	全体式 $\overline{P}$ から到達可能な全体式の集合

□

定義 3.5（個別式の充足関係）

$\forall P\in\Pi$	$P\models T$
$\forall P\in\Pi,\mathcal{A}\in\Phi^{-t}$	$P\models\neg\mathcal{A}\stackrel{\triangle}{=}\neg(P\models\mathcal{A})$
$\forall P\in\Pi,\mathcal{A},\mathcal{B}\in\Phi^{-t}$	$P\models\mathcal{A}\vee\mathcal{B}$
	$\stackrel{\triangle}{=}P\models\mathcal{A}\vee P\models\mathcal{B}$
$\forall P\in\Pi,\mathcal{A},\mathcal{B}\in\Phi^{-t}$	$P\models\mathcal{A}\wedge\mathcal{B}$
	$\stackrel{\triangle}{=}P\models\mathcal{A}\wedge P\models\mathcal{B}$
$\forall P\in\Pi,\mathcal{A},\mathcal{B}\in\Phi^{-t}$	$P\models\mathcal{A}\mid\mathcal{B}$
	$\stackrel{\triangle}{=}\exists P',P''\in\Pi.P\equiv P'\mid P''$
	$\wedge P'\models\mathcal{A}\wedge P''\models\mathcal{B}$
$\forall P\in\Pi,n\in\mathcal{N},\mathcal{A}\in\Phi^{-t}$	$P\models n[\mathcal{A}]$
	$\stackrel{\triangle}{=}\exists P'\in\Pi.$
	$P\equiv n[P']\wedge P'\models\mathcal{A}$
$\forall P\in\Pi,\mathcal{A}\in\Phi^{-t}$	$P\models\blacklozenge\mathcal{A}$
	$\stackrel{\triangle}{=}\exists P'\in\Pi.P\downarrow^*P'\wedge P'\models\mathcal{A}$
$\forall P\in\Pi,\mathcal{A},\mathcal{B}\in\Phi^{-t}$	$P\models\mathcal{A}\Rightarrow\mathcal{B}$
	$\stackrel{\triangle}{=}P\models\mathcal{A}\Rightarrow P\models\mathcal{B}$

□

直観的に $P\models\mathcal{A}\mid\mathcal{B}$ は，P が $\mathcal{A}$ という論理式を満たすプロセスと，$\mathcal{B}$ という論理式を満たすプロセスの，*composition* 演算子による並列合成で構成されているプロセス式であることを示している．$P\models n[\mathcal{A}]$ は，P が $\mathcal{A}$ という論理式を満たすプロセスを持つ n というアンビアントであることを示している．$P\models\blacklozenge\mathcal{A}$ は，P が持つ階層構造のどこかに $\mathcal{A}$ という論理式を満たすプロセスが存在することを示している．

本論文では，時間経過にかかわる性質は全体式に対してのみ検査する．したがって，時相論理式の意味は定義 3.6 において全体式の充足関係として定義される．定義 3.6 では，全体式 $\overline{P}$ は $(P_1,\ldots,P_n)$ の形をしているものとする．

[*1] ◇との識別を容易にするため，本論文では *somewhere modality* の記号に ◆ を用いる．

定義 3.6（全体式に対する充足関係）

$$
\begin{array}{ll}
\forall \overline{P} \in \overline{\Pi} & \overline{P} \models T \\
\forall \overline{P} \in \overline{\Pi}, \mathcal{A} \in \Phi & \overline{P} \models \neg\mathcal{A} \triangleq \neg(\overline{P} \models \mathcal{A}) \\
\forall \overline{P} \in \overline{\Pi}, \mathcal{A}, \mathcal{B} \in \Phi & \overline{P} \models \mathcal{A} \vee \mathcal{B} \triangleq \overline{P} \models \mathcal{A} \vee \overline{P} \models \mathcal{B} \\
\forall \overline{P} \in \overline{\Pi}, \mathcal{A}, \mathcal{B} \in \Phi & \overline{P} \models \mathcal{A} \wedge \mathcal{B} \triangleq \overline{P} \models \mathcal{A} \wedge \overline{P} \models \mathcal{B} \\
\forall \overline{P} \in \overline{\Pi}, \blacklozenge q \Rightarrow f \in \Phi^{-t} & \overline{P} \models \blacklozenge q \Rightarrow f \\
 & \triangleq \forall i (1 \leq i \leq n).(P_i \models \blacklozenge q) \\
 & \Rightarrow P_i \models f \\
\forall \overline{P} \in \overline{\Pi}, \mathcal{A} \in \Phi & \overline{P} \models \Diamond \mathcal{A} \\
 & \triangleq \exists \overline{P'} \in \mathcal{R}(\overline{P}).\overline{P'} \models \mathcal{A} \\
\forall \overline{P} \in \overline{\Pi}, \mathcal{A} \in \Phi & \overline{P} \models \Box \mathcal{A} \\
 & \triangleq \forall \overline{P'} \in \mathcal{R}(\overline{P}).\overline{P'} \models \mathcal{A} \qquad \Box
\end{array}
$$

定義 3.6 の「$\overline{P} \models \blacklozenge q \Rightarrow f$」は，「$\overline{P} \models \Diamond \mathcal{A}$」，「$\overline{P} \models \Box \mathcal{A}$」といった $\overline{P}$ による時相式の充足を判定する過程で，各 $\overline{P'}$ が非時相論理式を満たすかどうかを判定するためのものである．本論文では，全体式 $(P_1, \ldots, P_n)$ が満たすべき非時相論理式を，$\blacklozenge co[T] \Rightarrow f$ という形をした極大非時相式に限定する．これにより，あるコンテナ co についての性質を検査する際，$P_i \models \blacklozenge co$ が偽になる個別式，つまりコンテナ co には無関係な個別式は f とは関係なく真になるため，co に関係のある部分式 P_i に対してだけ $P_i \models f$ の真偽を判定すれば十分となる．この具体例を，次節の式 (1) で説明する．

3.2　物流システムが満たすべき性質を表す論理式

本節では，定義 3.1 で示した，物流システムの満たすべき性質 p1～p3 が，本論文で定義した限定的な Ambient Logic によって表現できることを示す．このことを，輸出港 *PORT_A* からいくつかの港を経由し輸入港 *PORT_B* にコンテナ co を輸送する物流システムを例に説明する．

- いつか必ず co は *PORT_B* に輸送される．

$$\Box\Diamond(\blacklozenge co[T] \Rightarrow \blacklozenge PORT\_B[CY[co[T] \mid T] \mid T]). \quad (1)$$

式 (1) の $CY[co[T] \mid T]$ の部分はコンテナヤード CY の中にコンテナ co が存在しており，また $co[T] \mid T$ でコンテナヤード CY の中に，さらにコンテナ co 以外のものが存在してもよいことを示している．$\blacklozenge PORT\_B[CY[co[T] \mid T] \mid T]$ は，プロセス式のどこかで $PORT\_B[CY[co[T] \mid T] \mid T]$ という階層構造が存在することを示している．この式の前に $\blacklozenge co[T] \Rightarrow$ をつけることにより，全体式 $(P_1, \ldots, P_n)$ の中で co アンビアントを持つ個別式 P_i に対してのみ上記の条件の成立を検査することになる．さらにこの式に $\Box\Diamond$ をつけることで，どのような遷移が行われたとしてもどこかでその状態が必ず成り立つことを表している．p1 の性質はコンテナが輸入港のコンテナヤードにあればよいので，式 (1) で表すことができる．

- *PORT_A*，*PORT_B* 以外では co の積み下しは行われない．

$$\begin{aligned}\Box(\blacklozenge co[T] \Rightarrow (&\neg\blacklozenge PORT\_A[SHIP[T] \mid T] \\ &\wedge \neg\blacklozenge PORT\_B[SHIP[T] \mid T] \\ &\Rightarrow \neg\blacklozenge(SHIP[T] \mid co[T]))).\end{aligned} \quad (2)$$

p2 の性質は，コンテナを積み込む直前または積み下した直後は $(SHIP[T] \mid co[T])$ が成り立ち，積み込み，積み下ろしができるのは，それぞれ輸出港と輸入港だけなので “輸出港，輸入港以外ではコンテナの積み下しは行われない” といい換えることができ式 (2) で表すことができる．

- *SHIP*（コンテナ船）には指定された co のみが積み込まれる．

$$\begin{aligned}\Box(\blacklozenge co[T] \Rightarrow (&\neg\blacklozenge PORT\_A[\blacklozenge co[T]] \\ &\wedge \neg\blacklozenge PORT\_B[\blacklozenge co[T]] \\ &\Rightarrow \blacklozenge(SHIP[co[T] \mid T]))).\end{aligned} \quad (3)$$

p3 の性質は，輸出港か輸入港のどこかにコンテナがない場合には必ず指定されたコンテナ船にコンテナが積み込まれていることを要求しているので式 (3) で表現できる．この 3 つの式で，物流システムの所期の性質を本論文で扱う限定的な Ambient Logic で表現できる．

4.　弱双模倣等価性を用いたプロセス族のモデル検査

多数の貨物を扱う物流システムを多重 Ambient Calculus で表現する場合，貨物の個数ぶんの個別式を記述する必要があり，モデル検査の際の状態空間爆発が問題となる．ここでは多数の個別式からなる全体式に対するモデル検査を，より少数の個別式からなる全体式のモデル検査に帰着させる方法について述べる．

4.1　弱双模倣等価性

文献 [7] では多重 Ambient Calculus の全体式（プロセス系）間の弱双模倣等価関係を定義している．ここでは比較の対象とする全体式を，共通する個別式を持つ，$\overline{P} = (P_1, \ldots, P_l, P_{l+1}, \ldots, P_m)$ と $\overline{Q} = (P_1, \ldots, P_l, Q_{l+1}, \ldots, Q_n)$ のような形であり，さらにすべての大域アンビアントはそれらの共通部分式 $P_1, \ldots, P_l$ のいずれかに現れる，すなわち $\overline{P}$, $\overline{Q}$ の両方で $A_G = \bigcup_{i=1}^{l} A_G^i$ が成り立つものに限定した場合の弱双模倣等価性について説明する．

2 つの全体式 $\overline{P}$，$\overline{Q}$ の振舞を比較するうえで，それぞれにおいて生起可能なイベントの集合，すなわちラベル集合を観測可能なもの $\mathcal{O}$ と不能なもの $\mathcal{O}^c$ に分け，大域アンビアントに関する遷移はすべて観測可能，すなわち $\mathcal{L}(\overline{P}) \cap \mathcal{L}(\overline{Q}) \supseteq \mathcal{O} \supseteq \mathcal{L}_G(\overline{P}) = \mathcal{L}_G(\overline{Q})$ である場合を考える．

多重 Ambient Calculus の全体式 $\overline{P}$ に対して，$\overline{P}$ から到達可能な全体式の集合を $\mathcal{R}(\overline{P})$ とし，$\mathcal{R}(\overline{P})$ と $\mathcal{R}(\overline{Q})$ 間の関係

$$B=\{((R_1,\ldots,R_m),(S_1,\ldots,S_n)) \mid R_i = S_i(1 \le i \le l)\}$$

を考える．ただし，l は $\overline{P}$，$\overline{Q}$ の共通個別式の数である．

定義 4.1（弱双模倣等価）
$\overline{P}, \overline{Q}$ と観測可能イベント（遷移ラベル）の集合 $\mathcal{O}$ に対して，B が以下の性質を満たすとき，関係 B は $\mathcal{O}$ に対して弱双模倣であるという．また，$\overline{P}$ と $\overline{Q}$ は観測可能イベント集合 $\mathcal{O}$ に対して弱双模倣等価であるといい，$\overline{P} \simeq_{\mathcal{O}} \overline{Q}$ と書く．
$(\overline{R},\overline{S}) \in B$ であるとき以下が成り立つ．ただし $\mathcal{O}^c$ は $\mathcal{O}$ の補集合を表すものとする．

(1) $\overline{R} \to_l \overline{T}(l \in \mathcal{O})$ ならば $\overline{S} \stackrel{*}{\to}_{l'} \overline{U}$ $(l' \in (\mathcal{O}^c)^* l (\mathcal{O}^c)^*)$ かつ $(\overline{T},\overline{U}) \in B$ なる $\overline{U}$ が存在する．
(2) $\overline{R} \to_l \overline{T}(l \in \mathcal{O}^c)$ ならば $\overline{S} \stackrel{*}{\to}_{l'} \overline{U}(l' \in (\mathcal{O}^c)^*)$ かつ $(\overline{T},\overline{U}) \in B$ なる $\overline{U}$ が存在する．
(3) $\overline{S} \to_l \overline{U}(l \in \mathcal{O})$ ならば $\overline{R} \stackrel{*}{\to}_{l'} \overline{T}$ $(l' \in (\mathcal{O}^c)^* l (\mathcal{O}^c)^*)$ かつ $(\overline{T},\overline{U}) \in B$ なる $\overline{T}$ が存在する．
(4) $\overline{S} \to_l \overline{U}(l \in \mathcal{O}^c)$ ならば $\overline{R} \stackrel{*}{\to}_{l'} \overline{T}(l' \in (\mathcal{O}^c)^*)$ かつ $(\overline{T},\overline{U}) \in B$ なる $\overline{T}$ が存在する．　□

$\overline{P} \simeq_{\mathcal{O}} \overline{Q}$ ならば，$\mathcal{O}$ のみが観測可能であるとしたときの $\overline{P}$ から実行可能な観測可能イベント系列の集合と，$\overline{Q}$ から実行可能な観測可能イベント系列の集合が一致する．

全体式間の弱双模倣等価性が，一定の条件を満たす個別式の追加について閉じていることを表す以下の性質が成り立つことが文献 [7] に示されている．

定理 4.2　$\overline{P} = (P_0,\ldots,P_m)$，$\overline{Q} = (P_0,\ldots,P_l, Q_{l+1},\ldots Q_n)$ とし，それらから P_0 を除去した全体式 $\overline{P^-} = (P_1,\ldots,P_m)$，$\overline{Q^-} = (P_1,\ldots,P_l,Q_{l+1},\ldots,Q_n)$ とし，それらの大域遷移ラベルの集合は同じとする．$\mathcal{L}_G(\overline{P}) \subseteq \mathcal{O} \subseteq \mathcal{L}(\overline{P^-}) \cap \mathcal{L}(\overline{Q^-})$ なる観測可能イベント集合 $\mathcal{O}$ について，$\overline{P^-} \simeq_{\mathcal{O}} \overline{Q^-}$ ならば $\overline{P} \simeq_{\mathcal{O} \cup \mathcal{L}_I^0} \overline{Q}$ が成り立つ．　□

また $\overline{P}$ と $\overline{Q}$ を比較する際の，観測可能イベント集合を $\mathcal{O}$ としても同じ性質が成り立つ．

4.2　不変式を用いたプロセス族のモデル検査

文献 [11] ではプロセス族 $\mathcal{F}$ 中のすべてのプロセスがある性質 f を満たすことの証明を，$\mathcal{F}$ の不変式（Invariant）$\mathcal{I}$ を見つけることと $\mathcal{I} \models f$ の証明に帰着させる方法について述べている．ここでは，多重 Ambient Calculus で，全体式間を比較する基準として弱双模倣等価性を用いた場合について説明する．

定義 4.3（不変式）　多重 Ambient Calculus の全体式の無限または有限集合 $\mathcal{F} = \{\overline{P}_{(1)}, \overline{P}_{(2)}, \ldots\}$ に対して，ある全体式 $\mathcal{I}$ が $\mathcal{F}$ 中の任意のプロセス $\overline{P}_{(i)}$ に対して $\mathcal{I} \simeq_{\mathcal{L}(\mathcal{I})} \overline{P}_{(i)}$ であるとき，$\mathcal{I}$ を $\mathcal{F}$ の不変式という．　□

ここで論理式 f に対して，$\simeq_{\mathcal{L}(\mathcal{I})}$ が f の真偽を保存する関係である，すなわち $\mathcal{I} \simeq_{\mathcal{L}(\mathcal{I})} \overline{P}_{(i)}$ ならば $\mathcal{I} \models f$ と $\overline{P}_{(i)} \models f$ が同値であることが保証されれば，$\overline{P}_{(i)} \models f$ のモデル検査を $\mathcal{I} \simeq_{\mathcal{L}(\mathcal{I})} \overline{P}_{(i)}$ と $\mathcal{I} \models f$ のモデル検査に帰着できる．

4.3　物流システムのモデル検査への適用

検査の対象とする全体式を以下のような形のものに限定し，極大非時相式がすべて $(\blacklozenge co[T]) \Rightarrow g$ の形式をしているような論理式 f を満たすかどうかの検査方法について述べる．

$$\overline{P} = (P_1, P_2, \ldots P_n)$$
$$P_i = SHIProute_i \quad (1 \le i \le N_0)$$
$$P_i = Invoice(c_i, S_k, D_k) \quad (N_{k-1} < i \le N_k)\ (1 \le k \le R)$$

ここで N_0 は輸送に用いる船の数，R は積荷港，荷降ろし港で分類した貨物の種別数，$N_k - N_{k-1}$ は k 番目の種別の貨物の数を表す．

ただし，以下では混乱のない範囲で個別式の番号，個別アンビアントの名前は適宜書き換えるものとする．

ここで R と N_0 を定数とし，$SHIProute_i$，S_k，D_k が具体的に与えられた場合を考えると，$\mathcal{F} = \{\overline{P} \mid N_k - N_{k-1} \ge 1\ (1 \le k \le R)\}$ を考えることができる．このとき $\mathcal{F}$ の不変式の候補としてすべての航路と，各種別の貨物を 1 つずつ含む全体式

$$\mathcal{I} = (SHIProute_1, \ldots, SHIProute_{N_0}, Invoice(c_1, S_1, D_1), \ldots, Invoice(c_R, S_R, D_R))$$

を考える．

4.3.1　不変式であることの確認

実際に $\mathcal{I}$ が $\mathcal{F}$ の不変式になっていることを以下の手順で確認する．

(1) $1 \le k \le R$ について，$\mathcal{I} \simeq_{\mathcal{L}(\mathcal{I})} (\mathcal{I}, Invoice(c, S_k, D_k))$ を確認する．
(2) $1 \le k \le R$ について，$P = Invoice(c, S_k, D_k)$ が $\mathcal{I}$ に含まれない大域遷移を持たない，すなわち $\mathcal{L}_G((\mathcal{I}, P)) = \mathcal{L}_G(\mathcal{I})$ であることを確認する．

以上が確認できれば，定理 4.2 と関係 $\simeq_{\mathcal{L}(\mathcal{I})}$ の推移性より，$\overline{P} \in \mathcal{F}$ なるすべての $\overline{P}$ について $\mathcal{I} \simeq_{\mathcal{L}(\mathcal{I})} \overline{P}$ であることが保証される．

ただし (2) については，$\mathcal{I}$ 中に P と同じ S_k と D_k を持つ $Invoice$ 式が含まれていることより，いま考えている $\mathcal{I}$，$\mathcal{F}$ に対しては確認の必要はない．

4.3.2　$\simeq_{\mathcal{L}(\mathcal{I})}$ における f の真偽の保存

一般的には，$\simeq_{\mathcal{L}(\mathcal{I})}$ は f の真偽を保存しない．ここでは，

$\simeq_{\mathcal{L}(\mathcal{I})}$ を $\mathcal{F}$ 中の全体式どうしの比較のみに用いること，f の極大非時相式が $\mathcal{I}$ のいずれかの個別式（P_{co} とする）で用いられる個別アンビアント co について $(\blacklozenge co[T]) \Rightarrow g$ のような形に限定されている場合について考える．

$\overline{P}, \overline{Q} \in \mathcal{F}$ について $\overline{P} \simeq_{\mathcal{L}(\mathcal{I})} \overline{Q}$ であるとし，弱双模倣等価であることを保証する $\mathcal{R}(\overline{P})$，$\mathcal{R}(\overline{Q})$ 間の関係を B とする．$(\overline{P}', \overline{Q}') \in B$ なる任意の $\overline{P}'$，$\overline{Q}'$ を考えると，それらは P_{co} から遷移した共通の個別式 P'_{co} を含む．また，$\overline{P}'$，$\overline{Q}'$ 中の P'_{co} 以外の個別式は co を含まないので $\blacklozenge co[T] \Rightarrow g$ は真となる．結局 $\overline{P}' \models \blacklozenge co[T] \Rightarrow g$，$\overline{Q}' \models \blacklozenge co[T] \Rightarrow g$ はともに $P'_{co} \models g$ に帰着され，$\overline{P}' \models \blacklozenge co[T] \Rightarrow g$ と $\overline{Q}' \models \blacklozenge co[T] \Rightarrow g$ の真偽は一致する．さらに，B が $\mathcal{L}(P_{co})$ を観測可能とした場合の弱双模倣関係であることより $\mathcal{R}(\overline{P})$，$\mathcal{R}(\overline{Q})$ 上での遷移関係について f の各極大非時相式の真偽の変動は一致することより，$\overline{P} \models f$ であるときかつそのときのみ $\overline{Q} \models f$ が導かれる．

4.3.3　モデル検査手順

以上の議論に基づいたモデル検査手順を以下に示す．

(1) 貨物の種別 k（$1 \leq k \leq R$）ごとに，コンテナアンビアントの名前 co_k として，その種別の貨物について満たすべき性質 $f(co_k)$ を記述し，$f(co_k)$ の極大非時相式が $(\blacklozenge co_k[T]) \Rightarrow g$ の形をしていることを確認する．
(2) 各 co_k について $\mathcal{I} \models f(co_k)$ であることを確認する．
(3) $\mathcal{I}$ が $\mathcal{F}$ の不変式になっていることを確認する．

以上により $\overline{P} \in \mathcal{F}$ なるすべての $\overline{P}$ について，$\overline{P} \models f(co_k)$ であることが保証される．

$\mathcal{I}$ 以外の個別式に現れる貨物 co の輸送に関する性質についても，co と同じ種別の貨物 co_k に関する性質 f を用いて $f(co)$ と記述できる．そして $\overline{P} \models f(co)$ は名前の付け変えにより $\overline{P} \models f(co_k)$ に変換できるので，$\overline{P} \models f(co_k)$ にあわせて $\overline{P} \models f(co)$ の成立も保証される．

5.　モデル検査システム

本研究では，文献 [6] のモデル検査システムをもとに，物流計画を多重 Ambient Calculus で記述したプロセス式のモデル検査，および弱双模倣等価性を検証するシステムを構築した．

5.1　検査システム

検査システムは Java で実装した．システムの規模は，クラス数 25，総行数 3,800 ほどである．このうち，文献 [6] の検査システムを，ほぼそのまま再利用したものが 11 クラス，多重 Ambient Calculus に対応させるため，変更したプログラムが 8 クラス，新たに構築したプログラムが 6 クラスとなっている．本検査システムでは，与えられた多重 Ambient Calculus で記述されたプロセス式に対して初めにプロセス式の遷移グラフを作成する．遷移グラフの各ノードは，ノードを識別するためのノード ID，遷移経路を知るための実行可能な遷移の情報と，それぞれの遷移経路から到達する子ノード ID，そして**図 1** に示すようなプロセスの全体式の構造を表す構文木のリストを持っている．構文木は各個別式ごとに分かれており，個別式には ID が与えられている．個別式内の構文木の各ノードは，アンビアントの名前，各アンビアントを個別式内で識別するための ID，親子関係などを識別するための親アンビアントの ID，*capability action* のリストを持っている．この構文木の *capability action* のリストと木の親子関係を見ることで可能な遷移を見つけ，構文木を遷移させることで遷移グラフの作成を行う．可能な遷移が複数ある場合に枝分かれが生じる．この遷移グラフを深さ優先で探査し，各ノードの全体式に対してアンビアントの非時相式の検査を木の親子関係を見ながら行い，*sometime modality* や *everytime modality* のような時間的な検査を経路を見ながら行う．

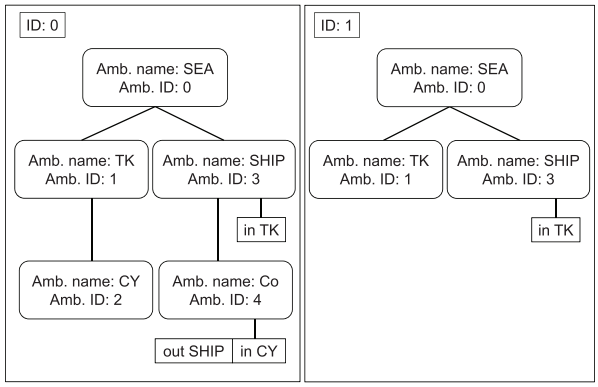

図 1　構文木
Fig. 1　A syntax tree.

5.2　検査方法

弱双模倣等価性の検査として，定義 4.1 の (1) と (2) に従って説明する．

(1) $(\overline{R}, \overline{S}) \in B$ に対して，$\overline{R}$ において実行可能な遷移 $l \in \mathcal{O}$ と，l で遷移した後の $\overline{T}$ を求める．次に，$\overline{S}$ において l が実行可能かどうか調べる．実行可能でなければ，$\mathcal{O}^c$ の遷移を繰り返し実行し l が実行可能な状態まで遷移させる．その後 l を実行させ，$(\overline{T}, \overline{U}) \in B$ となるような $\overline{U}$ が出現するまで $\mathcal{O}^c$ の遷移を繰り返す．
(2) $\overline{R}$ が実行可能な遷移 $l \in \mathcal{O}^c$ を持つ場合，l で遷移した後の $\overline{T}$ を求める．一方 $\overline{S}$ から $l' \in (\mathcal{O}^c)$ の遷移を $(\overline{T}, \overline{U}) \in B$ となる $\overline{U}$ が出現するまで繰り返して実行する．

5.3　検証実験

開発したモデル検査システムを用いて 定義 2.13 の形式の全体式を対象に，4.3.3 項に示した手順 (1)〜(3) に従って検証実験を行った．実験に使用した計算機のオペレーティングシステムは Vine Linux 5.2 64 bit 版，CPU

は Core-i7 3 GHz，メモリ容量は 8 GB である．

まず $N_0 = 1$，$R = 1$ の場合として $\mathcal{I}$ を

```
(Invoice(co1, TK, KB),
 SHIProute)
```

とした検証実験を行った．4.3.3 項の手順 (2) の $\mathcal{I} \models f(co1)$ の検証は 0.17 秒で行うことができた．また，4.3.3 項の手順 (3) における $\mathcal{I} \simeq_{\mathcal{L}(\mathcal{I})} (\mathcal{I}, Invoice(co2, TK, KB))$ の検証は 0.38 秒で行うことができた．$N_0 = 1$，$R = 1$ の場合，これらのモデル検査で東京港から神戸港へ任意の数の貨物を輸送する物流計画が所期の性質を満たしていることを確認したことになる．

次に，$N_0 = 1$，$R = 3$ の場合として $\mathcal{I}$ を

```
(Invoice(co1, TK, KB),
 Invoice(co2, KB, MJ),
 Invoice(co3, TK, MJ),
 SHIProute)
```

として検証実験を行った．手順 (2) の co1，co2，co3 に関する時相論理式の充足確認はそれぞれ 0.50 秒，0.51 秒，051 秒で完了した．手順 (3) における 3 つの弱双模倣等価性の検査はそれぞれ 4.48 秒，3.89 秒，4.26 秒で完了した．以上の結果を**表 1** に示す．

さらに，比較のため文献 [6] のモデル検査システムを用いて，$N_0 = 1$，$R = 1$ の場合および $N_0 = 1$，$R = 3$ の場合について貨物数を変動させた通常の Ambient Calculus のプロセス式を記述し，検証実験を行った結果を**表 2** に示す．

貨物数が貨物の種別数 R と同じ場合を除いて，本論文で提案した検証手順のほうが格段に検証時間を短縮できている．また文献 [6] のモデル検査システムでは $R = 1$ の場合には partial order reduction が有効に機能し状態空間爆発は抑制され，比較的多数の貨物を扱う物流計画を検証できているが，$R = 3$ の場合，対称性の検出が困難なため partial order reduction が機能していないことが分かる．一方，本論文で述べた検証システムによる実験では $R = 1$ の場合と $R = 3$ の場合にそれほど極端な差はなく，複数の種別の貨物が混在する現実的な物流計画のモデル検査に適用できるようになっている．

表 1　本論文のシステムによるモデル検査時間
Table 1　Verification time by the proposed model checker.

貨物の種類 (R)	1	3
手順 (2) の検査時間	0.17 秒	1.52 秒
手順 (3) の検査時間	0.38 秒	12.63 秒

表 2　文献 [6] のシステムによるモデル検査時間
Table 2　Verification time by the model checker in Ref. [6]

貨物の種類 (R)	1		
貨物数	1	10	100
検査時間	0.16 秒	853 秒	約 10 分

貨物の種類 (R)	3			
貨物数	3	6	8	9
検査時間	1.53 秒	24.2 秒	約 68 分	計測不能

6. 結論

本論文では，多数の貨物を扱う物流システムに対し所期の性質を満たしているかどうかを検査できるモデル検査システムを，多重 Ambient Calculus とその弱双模倣等価性を利用して構築した．本論文の主な成果は以下のようにまとめることができる．多重 Ambient Calculus によりモデル化された物流システムに対するモデル検査システムのための公理系の策定，多数のコンテナを運ぶような大規模な物流システムのモデル検査を，それと等価な少数のコンテナを運ぶ物流システムのモデルに対する検査に帰着させるための弱双模倣等価関係を用いた方法の提案，多重 Ambient Calculus のプロセス式（全体式）間の弱双模倣等価性を検証するシステムの構築，そしてそれらの成果を利用したモデル検査システムの構築および実証実験．

文献 [6] では既存の partial order reduction [11] やプロセス式の対称性を利用してモデル検査の状態空間爆発問題に取り組んだが，200 個程度のコンテナを輸送する物流システムのモデル検査が可能になるにとどまっていた．本論文で提案した新たな方法により，さらに多くの貨物を輸送する物流システムに対してもモデル検査が可能となることを確認できた．

今後は，貨物の乗せ換えを含むより複雑な経路，大規模な輸送などを対象としたモデル検査の実証実験を行う予定である．また本論文では，検査すべき性質を物流システムに対する所期の性質を表す論理式 3 種類に限定したが，現実世界では起こりえないことを記述していないことを表す論理式など，より多様な性質を検証できるようアルゴリズムを改善する必要がある．

謝辞　本研究は科研費（22500040）の助成を受けたものである．

参考文献

[1] 日本経済新聞社：海上コンテナの位置追跡，日本経済新聞 2007 年 1 月 24 日発行夕刊 3 面 (2007).
[2] 国土交通省：「メコン地域陸路実用化走行試験」—インドシナ半島物流を変える陸路物流の実用化へのチャレンジ，入手先 ⟨http://www.mlit.go.jp/kisha/kisha07/15/151018_.html⟩.
[3] 国土交通省：米国国土安全保障省及び国土交通省による海上貨物追跡タグシステム（MATTS）の通信能力実証実験，入手先 ⟨http://www.mlit.go.jp/kisha/kisha07/11/110427_.html⟩.
[4] Cardelli, L. and Gordon, A.D.: Mobile Ambients, *Theoretical Computer Science*, Vol.240, pp.177–213 (2000).
[5] 森本大輔，加藤　暢，樋口昌宏：Ambient Calculus を用いた物流検査システム，情報処理学会論文誌：プログラミング，Vol.48, No.SIG 10(PRO33), pp.151–164 (2007).

[6]　加藤　暢，樋口昌宏，植田直人：物流システムに対するAmbient Logic モデル検査システム，情報処理学会論文誌 数理モデル化と応用，Vol.3, No.1, pp.73–86 (2010).
[7]　樋口昌宏，加藤　暢：物流システム記述のための多重Ambient Calculus，情報処理学会論文誌 プログラミング，Vol.5, No.2, pp.79–87 (2012).
[8]　片山立志：よくわかる貿易書類入門，日本能率協会マネジメントセンター (2006).
[9]　商船三井ロジスティック：インボイス・パッキングリスト製作支援，入手先 ⟨http://www.mol-logistics.co.jp/japan/jp/support/invoice/sample_invoice.shtml⟩.
[10]　Cardelli, L. and Gordon, A.D.: Anytime Anywhere Modal Logics for Mobile Ambients, *Proc. 2000 ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pp.365–377 (2000).
[11]　Clarke, E.M., Grumberg, O. and Peled, D.A.: *Model Checking*, The MIT Press (1999).

樋口 昌宏　（正会員）

1983 年大阪大学基礎工学部情報工学科卒業．1985 年同大学院博士前期課程修了．（株）富士通研究所勤務，大阪大学基礎工学部助手・講師等を経て，現在，近畿大学理工学部情報学科准教授．博士（工学）．分散システムの記述，検証，試験に関する研究に従事．

森田 哲平　（正会員）

2010 年近畿大学理工学部情報学科卒業．2012 年同大学院総合理工学研究科エレクトロニクス系工学専攻博士前期課程修了．現在，（株）シー・エス・イー勤務．在学中，モデル検査に関する研究に従事．

加藤 暢　（正会員）

1997 年岡山大学大学院自然科学研究科博士課程修了．博士（工学）．1998 年日本学術振興会特別研究員（PD）．2000 年より近畿大学理工学部講師．現在，准教授．並行論理型言語の意味論，プロセス代数の研究に従事．