移动进程的蛰伏性及空间逻辑

林荣德[1,2] 奚建清[1] 郭玉彬[3]

(华南理工大学计算机科学与工程学院 广州 510640)[1] (华侨大学数学科学学院 泉州 362021)[2]

(华南农业大学信息学院 广州 510640)[3]

摘 要 界程逻辑(Ambient Logic)定义了一个示范性的空间逻辑来描述移动界程演算中移动进程的空间性质。然而在某些移动计算系统中,界程逻辑对移动进程空间性质的描述粒度是不够的。分析移动进程的蛰伏性质,用蛰伏和活跃来描述移动进程的存在状态,由此给出一种界程逻辑的扩展,称为状态空间逻辑。该逻辑能够描述移动进程的蛰伏性,进而更细粒度地刻画进程空间性质,且其在移动界程演算上的满足性是可判定的。同时还给出了状态空间逻辑公式的形式解释和蛰伏空间公式的逻辑推导规则。

关键词 移动界程演算,蛰伏进程,空间逻辑

Dormancy and Spatial Logic of Mobile Ambients

LIN Rong-de[1,2] XI Jian-qing[1] GUO Yu-bin[3]

(School of Computer Science and Engineering, South China University of Technology, Guangzhou 510640, China)[1]

(School of Mathematical Science, Huaqiao University, Quanzhou 362021, China)[2]

(College of Information, South China Agriculture University, Guangzhou 510640, China)[3]

Abstract Ambient logic presented a schematic spatial logic for specifying and reasoning about spatial properties of induced space of ambient processes. However, in some scenarios, spatial properties of processes should be specified and analyzed in fine-grained. Dormancy of mobile processes was analyzed, and their existence states can be described as dormant or active. Based on this, an extension of ambient logic, Spatial-state Logic, was presented. This logic can express dormancy of processes, thus spatial properties of processes can be specified in fine-grained. And satisfiability of the logic is decidable. Furthermore, formal interpretation over mobile ambients and inference rules of dormant spatial formulas were given.

Keywords Mobile ambients, Dormant process, Spatial logic

1 引言

移动界程演算[1-3](Mobile Ambients,简称界程演算),用"界程"的概念来刻画有边界的计算系统,如Web页面、组件、移动设备等。在界程演算中,界程是形如 $n[P]$ 的特殊进程。它表示名字为 n 、内部进程为 P 组成的计算区域。内部进程 P 又可以由其它界程组成。界程所表示的计算区域构成树型结构,且每个子树都可视为由该根节点所标识的子区域。近年来,移动界程演算已经应用于各种领域,如基于组件软件开发(CBSD)和面向软件方面开发(AOSD)的系统建模[4,19],Agent系统的安全性分析[5],主动式多媒体内容控制模型[6]和细胞膜演算[7]等方面。

移动界程演算及其扩展[9-11]都定义了两种类型的进程:一种是形如 $n[P]$ 的进程,其中 n 为命名边界,称之为界程[12];另一种是形如 $M.P$ 的带动作前缀的进程,称为动作进程;其中进程 P 可以是界程、动作进程或二者的组合。

为形式化描述移动界程演算的空间性质,文献[8]提出了界程逻辑(本文简称AL或AL逻辑)。AL逻辑有3个最基本空间公式:0,$n[\mathcal{A}]$和 $\mathcal{A}|\mathcal{B}$ 分别描述了空、界程以及两个任意进程的并行组合。AL逻辑公式能够直观地描述移动界程演算中界程的空间性质,但对动作进程的空间性质却需要用模态词 $\triangleright$ 来表达[20]。而模态词 $\triangleright$ 会导致AL逻辑公式满足性的不可判定[21]。另一方面,可判定的AL子逻辑(不包含模态词 $\triangleright$)对动作进程的性质描述却显得不够精细。比如,给定如下二个动作进程 P 和 Q,

$P=in\ n.(m[0]|\ open\ n.m'[0])$

$Q=out\ n.m'[0]$

直观上,P 和 Q 是两个差别很大的进程,但可判定的AL子逻辑只能用同样的公式:$\neg(\neg 0\ |\neg 0)$来描述,即在动作进程中出现的界程被忽略了。

在使用移动界程演算对移动计算系统进行建模时,在系统空间结构的性质及其随时间的变化需要特别关注的情景

到稿日期:2008-04-16 本文受国家教育部项目(D4109029),广东省科技计划项目(2006B11301001),广东省工业科技攻关计划项目(2006B80407001),广东省国际科技合作计划项目(2007A050100026)资助。

林荣德 男,在职博士生,主要研究领域为网络计算、形式化方法等,E-mail:linrd@hqu.edu.cn;奚建清 男,博士,教授,博士生导师,主要研究领域为信息与知识集成、网络计算等;郭玉彬 女,博士,讲师,主要研究领域为数据库与网络计算。

下，必须精确地描述动作进程的空间结构性质。

例 1　假定要设计一个消息传递系统，目标需求是："消息代理 *msg* 及其所包含的数据 *D* 被包装于 *mail* 中，开始时它们都位于主机 *send* 中，最终数据 *D* 能够到达主机 *receive*"。

将上述消息传递系统的目标需求描述（系统需求规范）采用可判定的 AL 公式描述如下：

$$send[mail[msg[D[T]]\,|\,T]]\,|\,receive[T]\wedge\Diamond(receive[D[T]]\,|\,T) \quad (1)$$

考虑一个使用移动界程演算作为建模工具描述的系统设计实例 $sys1 = Sender \mid Receiver$ 其中，

$Sender = send[mail[\text{out } send.\text{ in } receive \mid msg[D[0]]]]$

$Receiver = receive[open\ mail.\ open\ msg]$

通过分析 *sys*1 的归约过程可知，该设计实例满足目标需求。同时 *sys*1 也满足 AL 公式表示的系统需求规范 F1-1。

再考虑另一个系统设计实例 $sys2 = 2ndSender \mid Receiver$ 其中：

$2ndSender = send[mail[out\ send.\ in\ receive.\ msg[D[0]]]]$

$Receiver = receive[open\ mail.\ open\ msg]$

对 *sys*2 的归约过程进行分析可知，该设计实例也满足目标需求，即 *sys*2 与 *sys*1 对系统目标来说是行为等价的。然而 *sys*2 并不满足需求规范(1)。也就是说，*sys*1 和 *sys*2 之间的行为等价性没有在描述系统行为的逻辑公式(1)中反映出来。这表明公式(1)没有完整地描述满足目标需求的各种系统行为。其原因是公式(1)对动作进程 *out send. in receive. msg* [*D*[0]]的中界程 *msg* [*D*[0]]的空间性质无法进行描述。

为了细致地描述动作进程的空间结构和行为性质，本文分析了动作进程及其内部子进程归约行为的蛰伏性质，并用蛰伏和活跃来区分进程的存在状态；进而提出了一种可判定的界程逻辑扩展：即状态空间逻辑。该逻辑引入蛰伏模态来描述进程的蛰伏性质，可以更细粒度地描述动作进程的空间性质。文章还给出该逻辑在移动界程演算模型上的语义解释及蛰伏空间公式的推导规则。

本文的内容安排如下，第 2 节介绍移动界程演算的概念和移动进程的蛰伏性质；第 3 节定义状态空间逻辑公式和在移动空间模型上的逻辑语义；第 4 节给出一个使用状态空间逻辑作为移动计算系统建模的"需求规范"的实例；第 5 节别介绍了相关研究工作，最后总结全文。

2　相关概念

为了便于讨论移动界程演算所刻画的空间性质，本节给出移动界程演算的相关概念，详细内容可参考文献[1-3]，并基于此给出蛰伏进程定义。

2.1　移动界程演算概述

令 $\mathcal{N}$ 是名字的无穷可数集，其元素一般用 n,m 表示。

定义 1（动作）　动作是串行可执行代码，由下列 BNF 语法定义。

$$M,N ::= in\ n \mid out\ n \mid open\ n \mid M.\ N$$

上述定义中，动作包括了基本动作 *in n* ，*out n*，*open n* 和它们的串行组合。令 Σ 表示所有动作构成的集合，其元素一般由 M,N 来表示。

定义 2（移动进程，简称进程）　由下列 BNF 语法定义。

$$P,Q ::= 0 \mid P|Q \mid n[P] \mid M.\ P$$

其中，0 是一个特殊进程；$P|Q$ 是 P 和 Q 的并行组合进程；$n[P]$ 是名字 n 与 P 组合而成的界程，其中名字 n 表示进程 P 的位置(location)；$M.\ P$ 是动作 M 和进程 P 的串行组合进程，称动作进程。令 Π 表示所有进程构成的集合，其元素一般用 P,Q 表示。

进程 P 的名字集、自由名字集和绑定名字集分别用 $names(P)$，$fn(P)$ 和 $bn(P)$ 表示[1]；仅仅在绑定名字上不同的进程称为 α-等价的。本文将不区分 α-等价的进程。本文以下叙述中，界程 $n[0]$ 缩写为 $n[]$，进程 $M.\ 0$ 缩写为 M。

定义 3　移动进程的结构等价关系是，满足表 1 等价规则的移动进程集 Π 上的最小等价关系 $\equiv$。

表 1　移动进程的结构等价关系 $\equiv$

$P\equiv P$	Refl
$P\equiv Q \Rightarrow Q\equiv P$	Symm
$P\equiv Q \wedge Q\equiv R \Rightarrow P\equiv R$	Trans
$P\equiv Q \Rightarrow P\mid R\equiv Q\mid R$	Par
$P\equiv Q \Rightarrow n[P]\equiv n[Q]$	Ambient
$P\equiv Q \Rightarrow M.\ P\equiv M.\ Q$	Action
$P\mid Q\equiv Q\mid P$	Par Comm
$(P\mid Q)\mid R\equiv P\mid(Q\mid R)$	Par Assoc
$P\mid 0\equiv P$	Zero Par
$(M.\ N).\ P\equiv M.\ N.\ P$	Stru.

定义 4　进程包含关系 $\sqsubset\subset\Pi\times\Pi$ 定义如下：给定移动进程 $P,P'\in\Pi$，如果满足如下条件之一：

1）存在 $P''\in\Pi$ 且 $P''\neq 0$，$P\equiv P'\mid P''$；

2）存在 $n\in\mathcal{N}$，$P\equiv n[P']$

3）存在 $M\in\Sigma$，$P\equiv M.\ P'$。

则 $(P',P)\in\sqsubset$，即称 P' 是 P 的子进程，记为 $P'\sqsubset P$。

$\sqsubset^*$ 表示 $\sqsubset$ 的自反传递闭包关系。

定义 5　进程位置嵌套关系 $\downarrow\subset\Pi\times\Pi$，是一种特殊的进程包含关系，即 $\downarrow\subset\sqsubset$。给定移动进程 $P,Q\in\Pi$，如果，

$\exists P':\Pi, n:\mathcal{N}.\ P\equiv n[Q]\mid P'$

则 $(P,Q)\in\downarrow$，即称 Q 是 P 的直接嵌套，记为 $P\downarrow Q$。

$\downarrow^*$ 表示 $\downarrow$ 的自反、传递闭包，即 $P\downarrow^* Q$ 表示：Q 或者等于 P，或者被完全嵌套在 P 中的某个子界程中。

定义 6　移动界程演算归约系统是二元组 $\mathcal{TSP}=\langle\Pi,\rightarrow\rangle$，其中 Π 是移动进程集合，$\rightarrow\subset\Pi\times\Pi$ 是归约关系，由表 2 给出了进程归约规则来定义。给定 $P,Q\in\Pi$，用 $P\rightarrow Q$ 表示 $(P,Q)\in\rightarrow$。$\rightarrow^*$ 表示 $\rightarrow$ 的自反传递闭包关系。

表 2　移动界程演算的归约规则

$n[in\ m.\ P\mid Q]\mid m[R]\rightarrow m[n[P\mid Q]\mid R]$	Red In
$m[n[out\ m.\ P\mid Q]\mid R]\rightarrow n[P\mid Q]\mid m[R]$	Red Out
$Open\ n.\ P\mid n[Q]\rightarrow P\mid Q$	Red Open
$P\rightarrow Q \Rightarrow n[P]\rightarrow n[Q]$	Red Amb
$P\rightarrow Q \Rightarrow P\mid R\rightarrow Q\mid R$	Red Par
$P\equiv P', P\rightarrow Q, Q\equiv Q' \Rightarrow P'\rightarrow Q'$	Red $\equiv$

2.2　移动进程的蛰伏性

蛰伏性是一个生物学的概念，是对生物体活动的一种观察结果。用于描述某些生物体在一定环境下活动处于停止状态，一旦环境适宜则可重新变为活跃状态的生命特性。在计算机科学领域，蛰伏性也可以描述某个对象的计算过程的状态变化。比如在分布式操作系统环境下，活动进程在迁移到目标主机之前处于蛰伏状态，到达之后则可能处于活跃状态。

本文将蛰伏性引入到移动界程演算中，用于描述移动进程的归约行为。

例 2　设有动作进程 $P=in\ n.\ P'$ 和 $Q=in\ n.\ Q'$，其中子进程 P' 和 Q'：

$P'=m'[0]\mid open\ m'.\ n'\quad[0]$

$Q'=n'[0]$

根据定义 6 中的移动界程演算的归约关系，虽然子进程 $P'\rightarrow Q'$ 的归约是成立的，但 $P\rightarrow Q$ 的归约是不成立的，即

$P'\rightarrow Q'\not\Rightarrow in\ n.\ P'\rightarrow in\ n.\ Q'$

上例表明进程 P' 与动作 $in\ n$ 串行组合成为动作进程 $in\ n.\ P'$ 后，进程 P' 的归约处于阻塞状态。

性质 1(动作进程的蛰伏性)　在移动界程演算中，给定任意动作 M 和进程 P，Q，

$P\rightarrow Q\not\Rightarrow M.\ P\rightarrow M.\ Q$

证：因为由表 2 定义的移动界程演算归约规则无法导出这样的规则：$P\rightarrow Q\Rightarrow M.\ P\rightarrow M.\ Q$，所以该结论成立。

性质 1 指出形如 $M.\ P$ 的移动进程中所有子进程归约处于阻塞状态。而根据表 2 的进程归约规则，形如 $P\mid Q$ 和 $n[P]$ 的进程中的子进程归约未被阻塞。

定义 7(蛰伏进程和活跃进程)　给定移动进程 $P\in\Pi$，如果存在 $M\in\Sigma$ 和 $P'\in\Pi$，且 $P\equiv M.\ P'$，则 P 是一个蛰伏进程；否则 P 是一个活跃进程。

根据移动进程对子进程的归约行为是否阻塞来区分移动进程的存在状态(蛰伏的或活跃的)有利于细致地描述进程的行为。定义 7 从移动进程的组合结构上给出了不同类型进程的存在状态定义。

3　状态空间逻辑

状态空间逻辑(简称 SSL 逻辑)主要用于描述移动进程的空间结构、存在状态(活跃的还是蛰伏的)的性质及其随时间的变化。为了保证逻辑公式满足性的可判定性，SSL 逻辑没有采用模态词 $\triangleright$；同时新引入了蛰伏模态词 $\mathcal{O}$ 来描述进程的蛰伏性。比如：公式 $n[\psi]$ 描述的进程性质："一个活跃的，名称为 n 的界程且其内部进程性质为 ψ"，那么公式 $\mathcal{O}n[\psi]$ 描述的进程性质："一个蛰伏的，名称为 n 的界程且其内部进程性质为 ψ"。

3.1　SSL 的语法

蛰伏进程的一个重要特征是其内部子进程的所有归约行为都被阻塞，所以描述蛰伏进程内部性质的逻辑公式不应出现时态子公式。同时考虑到进程的相互组合关系，SSL 逻辑公式中时间模态不允许出现在描述进程空间结构的公式中。

定义 8　SSL 逻辑公式由表 3 中的 BNF 的语法给出。

直观上，SSL 公式由一阶谓词公式、时态公式和空间公式组成。其中时态公式是传统的一阶时态逻辑公式，它描述进程的空间结构和存在状态随时间的变化性质；空间公式 ψ 描述进程的空间结构和存在状态性质，包括活跃空间公式 α 和蛰伏空间公式 ξ，同时它们之间也是交替定义的：活跃空间公式 α 前加上蛰伏模态词 $\mathcal{O}$ 则成为蛰伏空间公式，活跃空间公式或蛰伏空间公式通过空间模态词的组合又成为活跃空间公式。这样的交替定义增强了空间公式的描述能力。比如，任意两个空间公式(活跃或蛰伏的)的组合 $\psi\mid\varphi$ 被定义成活跃空间公式，相应地，公式 $\mathcal{O}(\psi\mid\varphi)$ 是蛰伏空间公式。

表 3　SSL 逻辑公式

η 是一个名字 n 或一个名字变量 x
$\mathcal{A},\mathcal{B}::=$ SSL 公式
$\quad\neg\mathcal{A}\mid\mathcal{A}\vee\mathcal{B}\mid\forall x.\mathcal{A}\mid\Diamond\mathcal{A}\mid\psi$
$\psi,\varphi::=$空间公式
$\quad\neg\psi\mid\psi\vee\varphi\mid\xi\mid\alpha$
$\xi,\zeta::=$蛰伏空间公式
$\quad\neg\xi\mid\xi\vee\zeta\mid\mathcal{O}\alpha$
$\alpha,\beta::=$活跃空间公式
$\quad T\mid\neg\alpha\mid\alpha\vee\beta$
$\quad 0\mid\eta[\psi]\mid\psi@\eta\mid\psi\mid\varphi\mid\Diamond\psi$

下面给出一些派生的逻辑词的定义：$F\triangleq\neg T$，$\mathcal{A}\wedge\mathcal{B}\triangleq\neg(\neg\mathcal{A}\vee\neg\mathcal{B})$，$\mathcal{A}\Rightarrow\mathcal{B}\triangleq\neg\mathcal{A}\vee\mathcal{B}$，$\exists x.\mathcal{A}\triangleq\neg\forall x.\neg\mathcal{A}$。

给定一个 SSL 公式 $\mathcal{A}$ 用 $fn(\mathcal{A})$ 表示公式的自由名字集，用 $fv(\mathcal{A})$ 表示公式的自由变元集；其中对于蛰伏空间公式 $\mathcal{O}\alpha$：$fn(\mathcal{O}\alpha)=fn(\alpha)$ 和 $fv(\mathcal{O}\alpha)=fv(\alpha)$，而对于活跃空间公式 α：$fn(\alpha)$ 和 $fv(\alpha)$ 的定义与 AL 相同，可参考文献[1]。给定 SSL 公式 $\mathcal{A}$ 如果 $fv(\mathcal{A})=\emptyset$，则称公式 $\mathcal{A}$ 是闭的。

以下给出一些 SSL 空间公式及所描述性质的例子。

例 3　以下二个 SSL 的空间公式分别了描述蛰伏进程与活跃进程的并行组合和相互包含性质，

$$\psi=m[0]\mid\mathcal{O}n[0]\tag{2}$$

$$\psi=m[\mathcal{O}n[0]]\tag{3}$$

空间公式(2)描述："进程是由一个名称为 m 且其内部为空的界程和另一个'蛰伏'的，名称为 n 且其内部为空的界程的并行组合"。空间公式(3) 描述："进程是一个名称为 m 的界程，内部有一个'蛰伏'的，名称为 n，且其内部为空的界程"。

例 4　以下二个 SSL 公式描述了进程空间结构上的 somewhere 性质和进程的空间性质随时间的变化

$$\psi=\Diamond\mathcal{O}m[0]\tag{4}$$

$$\mathcal{A}=\Diamond\mathcal{O}\Diamond m[0]\tag{5}$$

空间公式(4)描述："进程或其所包含的某个子进程(直接或间接包含)，存在一个蛰伏的，名称为 m，且其内部为空的界程"。公式(5)描述："在某个时刻，进程是一个蛰伏的，其内部的某处存在一个名称为 m 且内部为空的界程"。

3.2　SSL 公式在移动界程演算模型上的满足关系

SSL 公式的满足关系 $P\vDash\mathcal{A}$ 表示了进程 P 满足闭公式 $\mathcal{A}$。以下首先来讨论蛰伏空间公式的满足性问题。

直观上，一个蛰伏空间公式 $\mathcal{O}\alpha$ 所描述的进程的空间性质对应了某个满足 α 性质的进程与动作串行组合后的性质，所以蛰伏空间公式的满足关系 $P\vDash\mathcal{O}\alpha$ 则意味着进程 P 经"展示"(去除动作)后的部分要满足活跃空间公式 α。因此蛰伏空间公式的满足关系可以形式化定义如下：

定义 9　给定移动进程 P 和蛰伏空间公式 $\mathcal{O}\alpha$，

$P\vDash\mathcal{O}\alpha$ 当且仅当，存在 $M,P',P\equiv M.\ P'$ 和 $Ex(P)\equiv P''$ 且 $P''\vDash\alpha$

其中"展示"函数 Ex 定义如下

$Ex(0)=0$

$Ex(M.\ P)=Ex(P)$

$Ex(n[P])=n[P]$

$$Ex(P|Q)=\begin{cases}Ex(P)|Ex(Q) & \text{if } P\equiv 0 \text{ or } Q\equiv 0\\ P\,|Q & \text{others}\end{cases}$$

上述定义的展示函数 Ex 对于每一个与动作组合的进程 $M.P$ 所计算的结果都是确定的，同时它有如下性质。

引理 1 如果 $P\equiv P'$，则 $Ex(M.P)\equiv Ex(M.P')$

证明：此结论可以根据定义 3 的移动界程演算的结构等价定义以及定义 2 所给出的移动界程的定义基础上，进行归纳证明。

定义 10(满足关系) 给定 $P\in\Pi$ 和 SSL 公式 $\mathscr{A}$，满足关系 $P\models\mathscr{A}$ 的归纳定义由表 4 给出。其中 Π 表示进程集合，$\mathcal{N}$ 表示所有名字集合，Σ 表示所有动作集合。

由表 4 中可知，SSL 的空间公式的满足性是定义在移动界程演算的结构等价基础上的，因此，如下性质。

命题 1 SSL 逻辑的空间公式的满足性在结构等价下保持不变，即

$\forall P,P',\psi.(P\models\psi\wedge P\equiv P')\Rightarrow P'\models\psi$

证明思路：可以根据移动进程结构等价定义和 SSL 公式结构上进行归纳得到。

表 4 SSL 公式在移动界程演算模型上的满足关系

$P\models T$
$P\models\neg\mathscr{A}\triangleq P\not\models\mathscr{A}$
$P\models\mathscr{A}\vee\mathscr{B}\triangleq P\models\mathscr{A}\vee P\models\mathscr{B}$
$P\models\forall x.A\triangleq\forall m:\mathcal{N}.P\models\mathscr{A}\{x\leftarrow m\}$
$P\models\Diamond\mathscr{A}\triangleq\exists P':\Pi.P\rightarrow^{*}P'\wedge P'\models A$
$P\models 0\triangleq P\equiv 0$
$P\models n[\psi]\triangleq\exists P':\Pi.P\equiv n[P']\wedge P'\models\psi$
$P\models\psi\mid\varphi\triangleq\exists P',P'':\Pi.P\equiv P'\mid P''\wedge P'\models\psi\wedge P''\models\varphi$
$P\models\Diamond\psi\triangleq\exists P':\Pi.P\downarrow{}^{*}P'\wedge P'\models\psi$
$P\models\psi@n\triangleq n[P]\models\psi$
$P\models\emptyset\alpha\triangleq\exists P':\Pi,M:\Sigma.P\equiv M.P'\wedge Ex(P)\models\alpha$

现在来讨论满足关系：$P\models\mathscr{A}$ 的可判定性问题。从文献[21]可知，检测一个不包含重复算子的移动界程演算模型和一个不包含模态词 $\triangleright$ 的闭公式之间的满足性是可判定的。SSL 逻辑并未使用并行模态词 $|$ 的对偶模态词 $\triangleright$，同时本文所考虑的移动空间演算也未包含重复算子。另外，对于 SSL 公式中的蛰伏模态，根据定义 9 和引理 1 可得蛰伏公式的满足性也是可判定的。因而有如下命题。

命题 2 给定移动进程 P 和 SSL 闭公式 $\mathscr{A}$，满足关系 $P\models\mathscr{A}$ 是可判定的。

3.3 SSL 的逻辑推导系统

如果一个闭公式能够被任一移动进程所满足则称该公式是永真的。本节将给出一些有关蛰伏空间公式的逻辑推导系统。需要指出是，AL 中的所有涉及空间公式的逻辑推导和推导规则都可以适用于 SSL 的活跃空间公式，限于篇幅本文不再列出，读者可查看文献[8]。

首先，在表 5 中定义了一些标记来分别表示 SSL 公式的逻辑推导系统。逻辑推导表示为一个前提和一个结果；而推导规则是用分号分隔的多个逻辑推导作为前提，来推导出另一个逻辑结果。

表 6 和表 7 列出有关蛰伏空间公式的推导规则，其中表 6 中的推导规则可视为逻辑定理，而表 7 中的派生规则可视为由表 6 中的推导规则得到的逻辑引理。

表 5 永真式、逻辑推导和推理规则标记定义

$vld(\psi)\triangleq\forall P:\Pi.P\models\psi$ 空间公式 ψ 是永真式
$\psi\vdash\varphi\triangleq vld(\psi\Rightarrow\varphi)$ 逻辑推导
$\psi\dashv\vdash\varphi\triangleq\psi\vdash\varphi\wedge\varphi\vdash\psi$ 逻辑等价
$\psi_1\vdash\varphi_1;\cdots;\psi_n\vdash\varphi_n\}\psi_0\vdash\varphi_0\triangleq$ 推导规则
$\psi_1\vdash\varphi_1\wedge\cdots\wedge\psi_n\vdash\varphi_n\Rightarrow\psi_0\vdash\varphi_0$
$\psi_1\vdash\varphi_1\{\ \}\psi_0\vdash\varphi_0\triangleq$ 双向推导
$\psi_1\vdash\varphi_1\}\psi_0\vdash\varphi_0\wedge\psi_0\vdash\varphi_0\}\psi_1\vdash\varphi_1$

表 6 蛰伏空间公式逻辑推导规则

$(\emptyset\rightarrow 0)$	$\}\emptyset\alpha\vdash\rightarrow 0$
$(\emptyset\rightarrow\mid)$	$\}\emptyset\alpha\vdash\rightarrow(\rightarrow 0\mid\rightarrow 0)$
$(\emptyset\wedge)$	$\}\emptyset\alpha\wedge\emptyset\beta\vdash\emptyset(\alpha\wedge\beta)$
$(\emptyset\vee)$	$\}\emptyset(\alpha\vee\beta)\vdash\emptyset\alpha\vee\emptyset\beta$
$(\emptyset\vdash)$	$\alpha\vdash\beta\{\ \}\emptyset\alpha\vdash\emptyset\beta$
$(\emptyset\rightarrow)$	$\}\emptyset\rightarrow\alpha\vdash\rightarrow\emptyset\alpha$

表 7 蛰伏模态公式的逻辑推论

$(\emptyset F)$	$\}\emptyset F\vdash F$
$(\emptyset\wedge)$	$\}\emptyset(\alpha\wedge\beta)\vdash\emptyset\alpha\wedge\emptyset\beta$
$(\emptyset\vee)$	$\}\emptyset\alpha\vee\emptyset\beta\vdash\emptyset(\alpha\vee\beta)$
$(\rightarrow\emptyset)$	$\}\rightarrow\emptyset\alpha\dashv\vdash\emptyset T\Rightarrow\emptyset\rightarrow\alpha$

表 6 所示的推导规则中，前两个规则分别表示任何一个蛰伏进程都不是空的和不可分解的。规则$(\emptyset\wedge)$和$(\emptyset\vee)$表示蛰伏模态词 $\emptyset$ 对逻辑连接词 $\wedge$ 和 $\vee$ 满足分配律。

规则$(\emptyset\vdash)$表示了，如果活跃进程集(分别由 α 和 β 代表)中存在包含关系，则当它们中的每一个进程都与任一动作组合后(即成为蛰伏进程后)，它们所对应的蛰伏进程集(分别由 $\emptyset\alpha$ 和 $\emptyset\beta$ 代表)仍然保持这种包含关系。反之亦然。这是一个很重要的推导规则，它表明每个从 AL 中派生过来的有关活跃空间公式的逻辑推导，都对应了一个蛰伏空间公式的逻辑推导。比如文献[8]中关于空间公式并行组合模态词的分配律规则$(\wedge\mid)$

$\}(\alpha\wedge\alpha')\mid\beta\vdash\alpha\mid\beta\wedge\alpha'\mid\beta$

则可以得到以下派生规则：

$\}\emptyset((\alpha\wedge\alpha')\mid\beta)\vdash\emptyset(\alpha\mid\beta\wedge\alpha'\mid\beta)$ 和

$\}\emptyset((\alpha\wedge\alpha')\mid\beta)\vdash\emptyset(\alpha\mid\beta)\wedge\emptyset(\alpha'\mid\beta)$。

规则$(\emptyset\rightarrow)$表示了公式 $\emptyset\rightarrow\alpha$ 所代表进程集是公式 $\rightarrow\emptyset\alpha$ 所代表的进程集的子集。因为前者代表了所有不满足 α 性质的进程与任一动作组合后的进程；而后者描述了除了满足 $\emptyset\alpha$ 性质的蛰伏进程之外的任何进程，即后者可以“蛰伏的”：不满足 α 的进程与任一动作组合后的进程(满足公式 $\emptyset\rightarrow\alpha$)；也可以是“活跃的”：满足 α 的活跃进程。

表 7 所示的逻辑推论中，推论$(\emptyset F)$表示了蛰伏模态只描述移动进程，而 F 不表示任何移动进程。推论$(\emptyset\wedge)$和$(\emptyset\vee)$表示了蛰伏模态词 $\emptyset$ 对逻辑连接词 $\wedge$ 和 $\vee$ 满足分配律是双向的。

推论$(\rightarrow\emptyset)$是规则$(\emptyset\rightarrow)$的直接结果，表示了如果只考虑蛰伏进程，则公式 $\rightarrow\emptyset\alpha$ 代表的进程与公式 $\emptyset\rightarrow\alpha$ 所代表的进程是等价的。

最后，需要说明是：本节给出 SSL 逻辑推导系统在移动界程演算模型上是语义一致的，但不是完全的。因为 SSL 注重的是提供一个观察角度来分析推理移动界程演算模型的进程状态，进程空间结构及随时间演变的性质，而不是去刻画移动界程演算模型的整个语义。这并不影响应用 SSL 公式来观察移动计算系统模型进程状态，进程空间结构及演变性质。

4 SSL 作为建模需求规范的实例

本节通过 SSL 公式来定义例 1 中移动计算系统建模的“需求规范”，来说明 SSL 公式通过描述进程的蛰伏性可以更细致地描述移动计算系统模型的行为。

接下来用 SSL 逻辑公式来形式化描述系统的目标需求。首先定义一个派生模态词◎如下：对于活跃空间公式 α，$\circledcirc\alpha \triangleq \alpha \vee \Diamond\alpha$。

则上述消息传递系统的目标需求描述（系统需求规范）可以用 SSL 逻辑公式形式化表示如下(6)。

$$(send[mail[\circledcirc msg[D[T]]|T]]|receive[T]) \wedge \Diamond(receive[D[T]]|T) \quad (6)$$

对 *sys*1 和 *sys*2 的归约过程进行分析可知，*sys*1 和 *sys*2 都满足公式(6)的系统目标规范，从而系统行为等价可以用逻辑性质的等价来刻画。因此，在移动计算系统建模过程中，一个子系统可以采用与之逻辑性质等价的其它子系统来替换，而不会影响整个系统的行为。

进一步地，根据公式(6)所表达的消息传递系统的系统需求规范，还可以得到满足需求规范(6)（即逻辑性质等价）的另外二个系统设计实例 $sys3 = 3rdSender|Receiver$ 和 $sys4 = 4thSender|Receiver$ 其中，

$3rdSender = send[mail[\ out\ send\ |\ in\ receive.\ msg\ [D[]]]\ ]$

$4thSender = send[mail[\ out\ send\ |\ in\ receive\ |\ msg\ [D[]]]\ ]$

$Receiver = receive[\ open\ mail.\ open\ msg\ ]$

上例表明，在利用移动界程演算进行移动计算系统建模时，SSL 逻辑公式能够更详尽地描述系统的“需求规范”。

5 相关研究

如上所述，L. Cardelli 和 A. D. Gordon 在文献[8]中提出了 AL 逻辑，同时文献[20]采用模态词 $\triangleright$ 来定义描述移动界程演算的动作进程的性质。

L. Cardelli 和 A. D. Gordon 还在文献[13]中对 AL 逻辑进行扩展以表达受限名字的性质。该文通过定义揭示名字公式 $n\textcircled{R}\mathcal{A}$ 来表示名字 n 仅受限于公式 $\mathcal{A}$ 所描述的进程内部，同时还分析了揭示名字公式语义与新鲜名字量词公式 $\nu x.\mathcal{A}$ 语义之间的关系；由此定义了隐藏名字公式 $(\forall x)\mathcal{A} \triangleq \nu x.\ x\textcircled{R}\mathcal{A}$ 来精确地描述移动界程演算中受限名字所表示的性质。

H. Lin[14,15]给出了基于谓词的移动界程空间逻辑，即谓词 μ-演算（Predicate μ-Calculus）。它通过谓词变量定义了不动点算符来表达公式的递归性，是对 AL 逻辑的一种扩展。

L. Caires 和 L. Cardelli[16,17]在异步 π-演算上提出的空间逻辑来表达 π-演算进程中的新鲜名字、受限名字、进程的空间结构和递归等性质。该逻辑除采用标准的一阶时态逻辑词外，还引入了进程组合模态、本地受限名量词、新鲜名量词以及递归操作符等。

M. Miculan 和 G. Bacci[18]提出了称为 Brane 逻辑的模态逻辑来描述生物学系统[7]的空间性质。Brane 逻辑的主要特点是能够表达动作的性质。它实际是由二个相互作用的子空间逻辑系统组成：一个用于描述“膜”（类似于移动界程演算中的动作）性质，另一个用于描述“系统”（类似于移动界程演算中的进程）性质。

上述工作，对 AL 逻辑都有不同程度的扩展，但都没有涉及到本文所讨论的蛰伏进程的性质。

结束语 本文分析了动作进程中其内部子进程归约行为的蛰伏性质，用蛰伏进程和活跃进程来描述移动进程的存在状态；提出了一种可判定的状态空间逻辑，并定义了一个蛰伏模态词来描述移动进程的蛰伏性质。最后给出了状态空间逻辑公式在移动界程演算模型上的形式解释以及蛰伏空间公式的逻辑推导规则。该逻辑提供了细粒度地描述移动计算系统建模中的进程空间结构，存在状态性质及随时间的变化性质的方法。

进一步的工作将采用 SSL 逻辑在面向方面的移动和分布式软件体系结构模型[4,19]中作为系统需求规范的工具，并研究用模型检测技术来自动验证软件模型的需求性质。

参 考 文 献

[1] Cardelli L, Gordon A D. Mobile ambients// Foundations of Software Science and Computation Structures: First International Conference, FOSSACS '98. Springer-Verlag, 1998

[2] Cardelli L, Gordon A D. Types for Mobile Ambients // Proc. POPL'99. ACM Press, 1999: 79-92

[3] Cardelli L, Gordon A D. Equational properties of Mobile Ambients. Mathematical Structures in Computer Science, 2003, 13 (3): 3712408

[4] Ali N, Millán C, Ramos I. Developing Mobile Ambients Using an Aspect-Oriented Software Architectural Model // Distributed Objects and Applications (DOA) 2006 International Conference. LNCS 4276. Berlin: Springer-Verlag, 2006: 1633-1649

[5] Nielson F, et al. Validating farewalls in mobile ambients// CONCUR'99, LNCS 1664. Springer, 1999: 463-477

[6] Tahara Y, Yoshioka N, Honiden S. A Formal Model of Active Contents Based on the Ambient Calculus// The 5th International Workshop on Mobile Agents for Telecommunication Applications, LNCS 2881. 2003: 132-141

[7] Cardelli L, Calculi B. In International Conference CMSB 2004. LNCS 3082. Springer-Verlag, 2005: 257-278

[8] Cardelli L, Gordon A D. Anytime, anywhere: Modal logics for mobile ambients// POLP 2000. ACM Press, 2000: 365-377

[9] Bugliesi M, Castagna G. Behavioural Typing for Safe Ambients. Computer Languages, 2002, 28(1): 61-99

[10] Bugliesi M, Castagna G, Crafa S. Boxed ambients // Theoretical Aspects of Computer Software 2001, LNCS 2215. Heidelberg: Springer Verlag, 2001: 38-63

[11] Fu Y. Fair ambients Acta Informatica. 2007, 43(8): 535-594

[12] Giovannetti E. Ambient Calculi with Types: a Tutorial// IST/FET International Workshop (GC 2003), LNCS 2874. Springer Berlin / Heidelberg, 2003: 151-191

[13] Cardelli L, Gordon A D. Logical properties of Name Restriction // Typed Lambda Calculi and Applications, LNCS 2004. Springer-Verlag, 2001

[14] Lin H. A predicate μ-calculus for mobile ambients and model checking// Proceedings of IWFMS 2004. 2004

[15] Lin H. A predicate spatial logic for mobile processes. Science in China Series F: Information Science, 2004, 47(3): 394-408

[16] Caires L, Cardelli L. A spatial logic for concurrency (Part I) // Proceedings of the 4th International Conference on Theoretical

Aspects of Computer Science (TACS 2001), LNCS 2215. Berlin: Springer Verlag, 2001: 1-37

[17] Caires L, Cardelli L. A spatial logic for concurrency (Part II)// Proceedings of the 13th International Conference on Concurrency Theory (CONCUR 2002), LNCS 2421. Berlin: Springer Verlag, 2002: 209-225

[18] Miculan M, Bacci G. Modal Logics for Brane Calculus// International Conference CMSB 2006. LNCS 4210. Springer-Verlag, 2006: 1-16

[19] Ali N, et al. Mobile Ambients in Aspect-oriented Software Architectures// Sacha K, ed. IFIP International Federation for Information Processing 2006. Volume 227, Software Engineering Techniques: Design for Quality. Boston: Springer, 2006: 37-48

[20] Hirschkoff D, Lozes E, Sangiorgi D. On the Expressiveness of the Ambient Logic. Logical Methods in Computer Science, 2006, 2(2)

[21] Charatonik W, et al. The Complexity of Model Checking Mobile Ambients// Honsell F, Miculan M, eds. Proceedings of the 4th International Conference on Foundations of Software Science and Computation Structures (FoSSaCS 2001). LNCS 2030. Berlin: Springer-Verlag, 2001: 152-167

(上接第 162 页)

既然分群可以使群体保持较好的多样性，那么不同的分群比是否可以作为调整群体多样性的一个策略呢？用 IPSO 方法对粒子分群比与种群多样性的关系进行分析，其中分群比采用粒子运动轨迹发散子群粒子数与局部搜索子群粒子数之比。可见，随着分群比的增加，种群熵值单调递增。图 2 显示了实验 2 条件下不同分群比 P 值条件下种群熵值的变化曲线。那么，是否存在一个合适的 P 值，使算法的跟踪效果更加有效呢？表 3 给出了在不同动态环境下 IPSO 方法的最佳分群比例 P，并且给出了此时的平均跟踪误差与种群熵值。

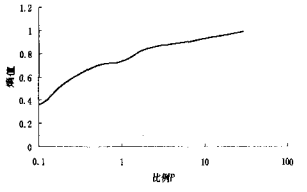

图 2　分群比与种群熵值的变化曲线

表 3　IPSO 方法的最佳分群比例及平均误差

指标	实验 1	实验 2	实验 3	实验 4	实验 5	实验 6
P 值	11/19	12/18	18/12	19/11	13/17	16/14
平均误差	0.016	0.061	0.042	0.0527	0.037	0.0428
熵值	0.671	0.691	0.837	0.849	0.704	0.811

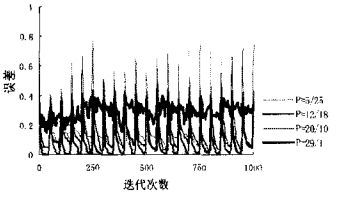

图 3　不同分群比 P 的跟踪误差效果

实验 1，2 从表 3 可以看到，在只有 1 个锥体的相对简单的动态环境下，当 P 取值在 0.5 左右时算法的跟踪效果较好，在这种环境下，算法不需要太多的分散粒子就能够很快跟踪到目标，此时种群的熵值稳定在 0.7 附近；而在 4 个锥体的相对复杂的动态环境下，P 的取值相对增大，介于[1.5，1.8]之间，也就是说，需要更多的发散粒子来快速找到新的动态最优点，而此时种群的熵值也相应增大。通过对比表 2 和表 3 可以看到，在不同动态环境下，通过适当选择分群比值 P，IPSO 方法的平均跟踪误差可以得到明显的降低。图 3 显示了实验二条件下 IPSO 方法在不同分群比值 P 时算法的跟踪效果。从图中可以看到，当 $P=12/18$ 时的跟踪效果是最好的；当 $P=20/10$ 时的跟踪效果要好于其它两种情况；当 $P=5/25$ 时，群体中 charged 个体相对较少，动态目标的变化情况对跟踪效果有很大影响。当 $P=29$ 时，也就是群体中只有一个 neural 个体，算法的跟踪效果反而较差，这是因为 charged 个体太多，使群体的趋同性下降，直接影响算法的跟踪效果。

结束语　本文采用种群熵来刻画粒子群算法中群体的多样性。在动态环境下对基本 PSO 方法、AutomicSwarm、PSOwDOW 和 IPSO 方法的多样性保持与算法的跟踪效果进行了对比，在此基础上通过研究不同的分群比对它们的影响，得到如下结论：

(1)动态环境下，群体多样性保持能够影响算法的跟踪效果，但这种影响并不完全是正向的；

(2)分群作为多样性保持的策略之一，可以通过调整分群比来达到改变群体多样性的目的；

(3)当环境变化幅度较大时，应调整分群比值使其大于 1，增加群体中发散个体的数目；而环境变化幅度较小时，分群比例值基本可选择小于 1，群体中发散个体的数目不必太多。

参 考 文 献

[1] Eberhart R C, Shi Y. Tracking and optimizing dynamic systems with particle swarms[A]// Proc. CEC 2001. NJ, 2001: 94-100

[2] 段晓东，王存睿，刘向东. 粒子群算法及其应用[M]. 沈阳：辽宁大学出版社，2007

[3] Hu Xiao hui, Eberhart R C. Adaptive Particle Swarm Optimization: Detection and Response to Dynamic Systems[A]. IEEE Congress on Evolutionary Computation. Honolulu, Hawaii, USA, 2002: 1666-1670

[4] Blackwell T M, Bentley P J. Dynamic Search with Charged Swarms[A]// Evolutionary Computation Conference. Honolulu, 2002: 19-26

[5] 窦全胜，周春光，等. 动态优化环境下的群核进化粒子群优化方法[J]. 计算机研究与发展，2006，43(1)：89-95

[6] 徐平，王存睿，段晓东，等. 基于粒子群的动态目标跟踪算法研究[J]，待发表

[7] 段晓东，高红霞. 基于种群熵和种群结构的粒子群算法研究[J]. 计算机工程与应用，2007，33(18)：222-224

[8] 段晓东，高红霞. 粒子群算法种群结构与种群多样性的关系研究[J]. 计算机科学，2007，34(11)：164-166

[9] Morrison R W, De Jong K A. A test problem generation for non-stationary environments[J]. IEEE Transactions on Evolutionary, 1999, 4: 2047-2053