

# Seal 演算的偶图语义

金龙飞 刘 磊

(吉林大学计算机科学与技术学院 长春 130012)

**摘 要** 偶图反应系统是一种新的理论工具,其基础是一种强调位置和连接的移动计算图形化模型——偶图,偶图范化了  $\pi$  演算和移动 Ambient 演算的特征,能够表示具有位置和移动性质的复杂系统. 偶图反应系统为普适计算不同层次的设计和实现提供了统一的建模框架. Seal 演算是一种用于描述移动计算的进程语言,具有良好的安全性质. 文中给出了一种不带复制进程表达式的 Seal 演算的偶图表示,分析了该 Seal 演算与其偶图表示间的结构对应和操作对应. 本研究扩展了偶图理论的应用范围,展示了偶图理论在描述安全演算方面的能力,为在偶图反应系统框架下研究 Seal 演算的性质和应用奠定了基础.

**关键词** Seal 演算;偶图;偶图反应系统;结构对应;操作对应  
**中图法分类号** TP311

## Bigraphical Semantics of Seal Calculus

JIN Long-Fei LIU Lei

(College of Computer Science and Technology, Jilin University, Changchun 130012)

**Abstract** Bigraphical Reactive Systems (BRS) is a new theoretical tool, and is based on a graphical model of mobile computation that emphasizes both locality and connectivity, named Bigraph. Bigraphs generalize both characteristics of the  $\pi$  calculus and the Mobile Ambient calculus, and it can represent complex systems that emphasize both locality and connectivity. The theory of BRS provides a uniform modeling framework for design and implementation of pervasive computing in different levels. The Seal calculus is a process language for describing mobile computation with security features. A bigraphical representation of a Seal calculus without replication is provided in this paper, and structural correspondence and operational correspondence between the Seal calculus and its bigraphical representation are analyzed. This work extends the application domains of BRS and illustrates the ability of BRS in describing security calculi. All are foundations for researching characteristics and applications of Seal calculus within the bigraphical framework.

**Keywords** Seal calculus; bigraph; bigraphical reactive system; structural correspondence; operational correspondence

## 1 引 言

Seal 演算是一种用于描述移动计算的进程语言<sup>[1]</sup>. Seal 演算在  $\pi$  演算的内核上扩展了同步机制

和客观移动机制. 与其他分布式计算模型如  $\pi$  演算、Ambient 演算相比,Seal 演算具有一些特殊的安全性质,如不可销毁的边界、父子交互等,因此非常适于作为探索移动安全性质的理论框架.

偶图反应系统 (Bigraphical Reactive System,

BRS)<sup>[2]</sup>是 Milner( $\pi$  演算的发明人)等提出的一种新的理论工具,其基础是一种强调位置和连接的移动计算图形化模型——偶图(bigraph). 一个偶图包括一个表示计算结点位置的位置图(place graph)和一个表示这些结点间连接关系的链接图(link graph),偶图的动态是用重写偶图的反应规则(reaction rule)来表示的,一个 BRS 就是反应规则的一个集合. BRS 的描述能力非常强,可以描述具有位置和连接性质的各种系统. BRS 理论有两个主要目的:(1)为普适计算(pervasive computing)不同层次的设计和实现提供统一的建模框架;(2)提供用于统一现有并发和移动演算的元理论,为各种进程演算提供统一的理论基础. Petri 网<sup>[3]</sup>、 $\pi$  演算<sup>[2]</sup>、CCS<sup>[4]</sup>、移动 Ambient、HOMER<sup>[5]</sup>和  $\lambda$  演算<sup>[6]</sup>已经被描述成 BRS. 另外,偶图理论也被用于描述上下文敏感(context-aware)系统<sup>[7]</sup>.

本文使用 BRS 作为元理论,根据 Seal 演算的特点,给出了一种不带复制进程表达式的 Seal 演算的偶图表示'*Seal*',分析了该 Seal 演算与'*Seal*'间的结构对应和操作对应. 本研究扩展了偶图理论的应用范围,展示了偶图理论在描述安全演算方面的能力,为在偶图反应系统框架下研究 Seal 演算的性质和应用奠定了基础.

本文第 2 节简单介绍偶图和偶图反应系统以及 Seal 演算的基本内容;第 3 节从结构(即静态)和操作(即动态)两个方面给出 Seal 演算的偶图表示;第 4 节介绍相关工作;第 5 节总结全文并讨论进一步的工作.

## 2 基础知识

### 2.1 偶图简介

偶图反应系统理论的核心是一种用于描述移动分布式 Agent 的拓扑图元模型——偶图,该模型可以描述 Agent 的连接和嵌套位置. 一个偶图包含两个结构:位置图和链接图. 位置图是一组无序的树,用于表示系统的拓扑结构. 树结构的根被称为区域(region),树的结点被称为位置(place),用于表示位置或进程构造子,如前缀等. 树的一些叶子被称为站点(site,也称为孔(hole)),通过这些站点使偶图成为一个(多孔的)上下文(context). 每个非站点的位置都有一个控制(control)来表示类型,每种控制都有一定数量的端口(port)来和链接图相连. 链接图表示系统的连接性,与  $\pi$  演算中的共享名字相对应. 自由名字表示为连接到偶图外部接口名字的链接.

偶图有许多不同的变体:纯偶图(pure bigraph)<sup>[4]</sup>中位置图和链接图是正交的,因而位置和连接没有相互关系,纯偶图可以表示纯的移动 Ambient 演算;在绑定偶图(binding bigraph)和局部偶图(local bigraph)<sup>[6]</sup>中这种正交性被打破了,绑定偶图可以表示  $\pi$  演算,局部偶图可以表示  $\lambda$  演算.

例如: $\pi$  演算的反应规则

$$\bar{x}y \mid x(z).P \rightarrow \{y/z\}P$$

可以表示成如图 1 所示的偶图反应规则. 其形式化表示为

$$send_{xy} \mid get_{x(z)} \square \rightarrow x \mid y/(z) \square.$$

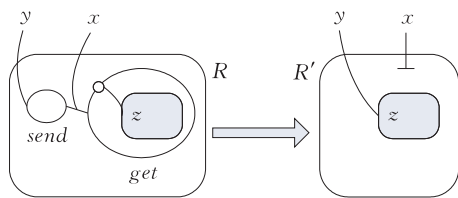


图 1 一个偶图的例子

篇幅所限,这里无法给出偶图和偶图反应系统完整的形式化描述,相关内容请参阅文献[2-4,6,8].

### 2.2 Seal 演算简介

Seal 演算是一种用于描述移动计算的进程语言. 与  $\pi$  演算相比,Seal 演算中线程和资源是树型结构的;与 Ambient 演算相比,Seal 演算中名字和 seal 间的联系较弱,seal 的边界是不可销毁的,并且 Seal 演算使用客观移动而不是 Ambient 中的主观移动. 本文考虑的是一种不带复制进程表达式的 Seal 演算,下面分别给出了这种 Seal 演算的语法、结构同余和反应规则.

#### 语法.

Names:

$$a, \dots, u, v, x, y, z, \dots \in N$$

Locations:

$$\begin{array}{ll} \eta ::= * & \text{local} \\ \mid \uparrow & \text{up} \\ \mid z & \text{down} \end{array}$$

Actions:

$$\begin{array}{ll} \alpha ::= \bar{x}^{\eta}(y_1, y_2, \dots, y_n) & \text{output} \\ \mid x^{\eta}(y_1, y_2, \dots, y_n) & \text{input} \\ \mid \bar{x}^{\eta}\{y\} & \text{send} \\ \mid x^{\eta}\{y_1, y_2, \dots, y_n\} & \text{receive} \end{array} \quad \text{for } n \geq 0$$

Processes:

$$\begin{array}{ll} P, Q, R ::= 0 & \text{inactivity} \\ \mid P \mid Q & \text{parallel composition} \\ \mid (\nu n)P & \text{restriction} \\ \mid a.P & \text{prefixing} \\ \mid n[P] & \text{seal} \end{array}$$

其中,  $N$  为名字的无穷集,  $0$  表示惰性进程, 进程  $\alpha.P$  由动作  $\alpha$  和接续进程  $P$  组成, 表示一个进程等待执行  $\alpha$  然后变成进程  $P$ ,  $P|Q$  表示由两个平行进程  $P$  和  $Q$  组成的进程,  $a[P]$  表示一个 seal, 进程  $P$  运行在  $a$  中, seal 本身也是一个进程, 可以嵌套在其它 seal 中.  $(\nu n)P$  表示  $n$  是  $P$  中的一个非自由变量.  $*$  表示通信通道在当前 seal 中,  $\uparrow$  表示通道在父 seal 中,  $z$  表示通道在子 seal  $z$  中. 动作 output 和 input 表示通信, 动作 send 和 receive 表示移动.

#### 结构同余.

$$\begin{aligned} P|0 &\equiv P & P|Q &\equiv Q|P & P|(Q|R) &\equiv (P|Q)|R \\ (\nu n)0 &\equiv 0 & (\nu n)(\nu m)P &\equiv (\nu m)(\nu n)P & n \neq m \\ (\nu n)(P|Q) &\equiv P|(\nu n)Q & n &\notin fn(P) \end{aligned}$$

结构同余是由静态上下文所保持的进程间的最小关系, 体现了进程间的结构等价.

#### 反应规则.

write local:

$$x^*(u).P|\bar{x}^*(v).Q \rightarrow P[v/u]|Q$$

write in:

$$\begin{aligned} \bar{x}^{?_1}(w).P|y[(\nu z)x^{?_2}(u).Q_1|Q_2] \\ \rightarrow P|y[(\nu z)(Q_1[w/u])|Q_2] \end{aligned}$$

write out:

$$\begin{aligned} x^{?_1}(u).P|y[(\nu z)(\bar{x}^{?_2}(v).Q_1)|Q_2] \\ \rightarrow (\nu v \cap z)(P[v/u]|y[(\nu z \setminus v)(Q_1|Q_2)]) \end{aligned}$$

move local:

$$\begin{aligned} x^*\{u\}.P_1|\bar{x}^*\{v\}.P_2|v[Q] \\ \rightarrow P_1|u_1[Q]|\dots|u_n[Q]|P_2 \end{aligned}$$

move in:

$$\begin{aligned} \bar{x}^{?_1}\{v\}.P|v[S]|y[(\nu z)(x^{?_2}\{u\}.Q_1|Q_2)] \\ \rightarrow P|y[(\nu z)(Q_1|Q_2|u_1[S]|\dots|u_n[S])] \end{aligned}$$

move out:

$$\begin{aligned} x^{?_1}\{u\}.P|y[(\nu z)(\bar{x}^{?_2}\{v\}.Q_1)|v[R]|Q_2] \\ \rightarrow P|(\nu fn(R) \cap z)(u_1[R]|\dots|u_n[R] \\ |y[(\nu z \setminus fn(R))(Q_1|Q_2)]) \end{aligned}$$

上述规则中, 函数  $fn$  为表达式的自由名字.  $P[v/u]$  表示用向量  $v$  去替换进程  $P$  中所有自由出现的向量  $u$  而得到的进程. 更详细的关于 Seal 演算的内容请参阅文献[1].

### 3 Seal 演算的偶图语义

为了描述方便, 规定如下:  $m$  和  $n$  表示名字的无穷集  $N$  中的元素;  $\bar{n}$  表示名字的有穷集中的元素;  $x$  和  $y$  表示进程变量的无穷集  $V$  中的元素;  $\bar{m}\bar{n}$  表示  $\bar{m} \cup \bar{n}$ , 其中  $\bar{m} \cap \bar{n} = \emptyset$ .

#### 3.1 Seal 演算的类型规则

为了给出 Seal 演算的偶图语义, 我们首先给出用于进程表达式类型断定的类型规则, 如表 4 所示, 类型规则形如  $\bar{x} \vdash P; \bar{n}$ , 表示  $P$  是类型良好的, 当且仅当  $P$  的自由名字和变量分别包含于集合  $\bar{n}$  和  $\bar{x}$ .

##### 类型规则.

$$\begin{aligned} &\frac{}{\bar{x} \vdash 0; \bar{n}} & \frac{\bar{x} \vdash P; \bar{n}_1 \quad \bar{x} \vdash Q; \bar{n}_2}{\bar{x} \vdash P|Q; \bar{n}_1 \cup \bar{n}_2} & \frac{\bar{x} \vdash P; \bar{n}n}{\bar{x} \vdash (\nu n)P; \bar{n}} \\ &\frac{\bar{x} \vdash P; \bar{n}}{\bar{x} \vdash n[P]; \bar{n}n} & \frac{\bar{x} \vdash P; \bar{n}}{\bar{x} \vdash \bar{x}^?(y).P; \bar{n} \cup \{x\} \cup y \cup fn(\eta)} \\ &\frac{\bar{x}y \vdash P; \bar{n}}{\bar{x} \vdash x^?(y).P; \bar{n} \cup \{x\} \cup fn(\eta)} \\ &\frac{\bar{x} \vdash P; \bar{n}}{\bar{x} \vdash \bar{x}^?(y).P; \bar{n} \cup \{x\} \cup \{y\} \cup fn(\eta)} \\ &\frac{\bar{x} \vdash P; \bar{n}}{\bar{x} \vdash x^?(y).P; \bar{n} \cup \{x\} \cup y \cup fn(\eta)} \end{aligned}$$

#### 3.2 基本表示

Seal 演算的偶图表示 'Seal 中用到的控制如表 1 所示, 其中控制的种类是由 Seal 演算的类型规则决定的.

表 1 'Seal 中的控制

控制	活跃	序数	功能
Output	被动	0→1+n 0→2+n	发送数据, 对应于 output 前缀
Input	被动	n→1 n→2	接收数据, 对应于 input 前缀
Send	被动	2→0 3→0	发送 seal, 对应于 send 前缀
Receive	被动	0→1+n 0→2+n	接收 seal, 对应于 receive 前缀
Seal	主动	0→1	seal 结构
Ann	被动	0→0	seal 类型标注
Tname	原子	0→1	seal 类型名字

由于前 4 种控制有多种序数, 因此严格地说, 它们是 4 个控制族, 当通过本地通道通信时, 一个端口连接到通道, 另  $n$  个 (Send 时为 1 个) 端口连接到数据; 当使用非本体通道通信时, 还要有一个端口连接到通道所在 seal 上. 'Seal 中 4 个控制族和 3 种控制的偶图如图 2 所示.

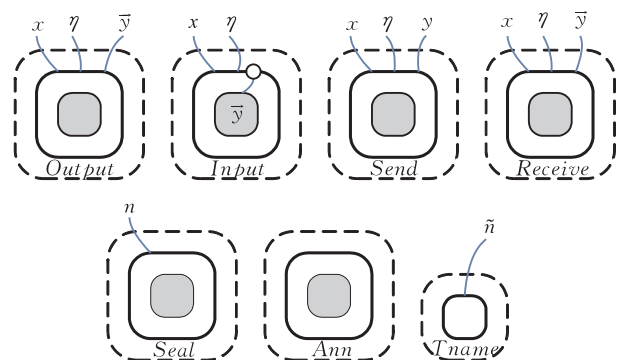


图 2 'Seal 中控制的偶图表示

### 3.3 转换规则

将 Seal 演算的类型规则转换到偶图术语的转换规则如下所示, 其中  $\llbracket \cdot \rrbracket$  是转换函数:

#### 转换规则.

$$\begin{aligned}
 \llbracket \tilde{x} \vdash 0; \tilde{n} \rrbracket &= \tilde{x} \oplus \tilde{n} \\
 \llbracket \tilde{x} \vdash P \mid Q; \tilde{n}_1 \cup \tilde{n}_2 \rrbracket &= \llbracket \tilde{x} \vdash P; \tilde{n}_1 \rrbracket \mid \llbracket \tilde{x} \vdash Q; \tilde{n}_2 \rrbracket \\
 \llbracket \tilde{x} \vdash (\nu n) P; \tilde{n} \rrbracket &= /n \circ \llbracket \tilde{x} \vdash P; \tilde{n} n \rrbracket \\
 \llbracket \tilde{x} \vdash n[P]; \tilde{n} n \rrbracket &= \\
 & \quad (Seal_n \oplus id_{\tilde{n} x}) \circ (\llbracket \tilde{x} \vdash P; \tilde{n} \rrbracket \mid (Ann \oplus id_{\tilde{n}}) \circ \llbracket \tilde{n} \rrbracket) \\
 \llbracket \tilde{x} \vdash \tilde{x}^y(y).P; \tilde{n} \cup \{x\} \cup y \cup fn(\eta) \rrbracket &= \\
 & \quad (Output_{fn(x^y)y} \oplus id_{\tilde{n} x}) \circ \llbracket \tilde{x} \vdash P; \tilde{n} \rrbracket \\
 \llbracket \tilde{x} \vdash \tilde{x}^y(y).P; \tilde{n} \cup \{x\} \cup fn(\eta) \rrbracket &= \\
 & \quad (Input_{fn(x^y)(y)} \oplus id_{\tilde{n} x}) \circ \llbracket \tilde{x} \vdash P; \tilde{n} \rrbracket \\
 \llbracket \tilde{x} \vdash \tilde{x}^y\{y\}.P; \tilde{n} \cup \{x\} \cup \{y\} \cup fn(\eta) \rrbracket &= \\
 & \quad (Send_{fn(x^y)y} \oplus id_{\tilde{n} x}) \circ \llbracket \tilde{x} \vdash P; \tilde{n} \rrbracket \\
 \llbracket \tilde{x} \vdash \tilde{x}^y\{y\}.P; \tilde{n} \cup \{x\} \cup y \cup fn(\eta) \rrbracket &= \\
 & \quad (Receive_{fn(x^y)y} \oplus id_{\tilde{n} x}) \circ \llbracket \tilde{x} \vdash P; \tilde{n} \rrbracket \\
 \llbracket \tilde{n} \rrbracket &= \bigcup_{n \in \tilde{n}} Tname_n
 \end{aligned}$$

为了简化, 这里的  $\oplus$  表示用配线 (wiring) 同时扩展一个偶图的内部接口和外部接口. 相关的说明和推导请参阅文献[5].

### 3.4 结构对应

**引理 1.** 如果  $\tilde{x} \vdash P \equiv Q; \tilde{n}$ , 则  $\llbracket \tilde{x} \vdash P; \tilde{n} \rrbracket = \llbracket \tilde{x} \vdash Q; \tilde{n} \rrbracket$ .

证明. 从 Seal 演算到  $'Seal$  的转换是可复合的, 因此我们分别考虑用  $\equiv$  定义的每个公理.

(1) 对于公理  $\tilde{x} \vdash P \mid 0 \equiv P; \tilde{n}$ ,  $\tilde{x} \vdash P \mid Q \equiv Q \mid P; \tilde{n}$  和  $\tilde{x} \vdash P \mid (Q \mid R) \equiv (P \mid Q) \mid R; \tilde{n}$ , 由于我们将 Seal 演算中的并行组合转换成偶图中的质积“ $\mid$ ”, 并且质积满足交换律和结合律, 同时由于 0 是质积的幺元, 所以我们可以从转换规则中直接得证.

(2) 对于公理  $\tilde{x} \vdash (\nu n)(\nu m) P \equiv (\nu m)(\nu n) P$ ;  $\tilde{n} n \neq m$ , 我们可以用下面相同的方法生成  $\llbracket x \vdash (\nu n)(\nu m) P; \tilde{n} \rrbracket$  和  $\llbracket x \vdash (\nu m)(\nu n) P; \tilde{n} \rrbracket$  对应的偶图, 因而得证: 首先构造  $\llbracket \tilde{x} \vdash P; \tilde{n} n m \rrbracket$ , 再用闭包配线  $/n$  和  $/m$  分别移除其外部名字  $n$  和  $m$ , 由于  $n \neq m$ , 所以按不同顺序移除而得到的两个偶图是同构的.

(3) 对于公理  $\tilde{x} \vdash (\nu n)(P \mid Q) \equiv P \mid (\nu n) Q; \tilde{n} n \notin fn(P)$ , 假设  $\tilde{n} = \tilde{n}_1 \cup \tilde{n}_2$ , 则  $\tilde{n}_1 \tilde{x}$  和  $\tilde{n}_2 n \tilde{x}$  分别是偶图  $\llbracket \tilde{x} \vdash P; \tilde{n}_1 \rrbracket$  和  $\llbracket \tilde{x} \vdash Q; \tilde{n}_2 n \rrbracket$  的外部名字, 将这两个偶图用质积组合起来, 然后用闭包配线  $/n$  移除其外部名字  $n$ , 则可得到  $\llbracket x \vdash (\nu n)(P \mid Q); \tilde{n} \rrbracket$  和  $\llbracket x \vdash P \mid (\nu n) Q; \tilde{n} \rrbracket$ , 因为  $n \notin fn(P)$ , 所以闭包配线只对  $\llbracket \tilde{x} \vdash Q; \tilde{n}_2 n \rrbracket$  起作用, 两种表示的偶图也是同构的.

(4) 对于公理  $\tilde{x} \vdash (\nu n) 0 \equiv 0; \tilde{n}$ , 由于  $n$  不是  $\llbracket \tilde{x} \vdash 0; \tilde{n} \rrbracket$  的外部名字, 因此带闭包配线  $/n \circ \llbracket \tilde{x} \vdash 0; \tilde{n} \rrbracket$  和不带闭包配线的  $\llbracket \tilde{x} \vdash 0; \tilde{n} \rrbracket$  是同构的. 证毕.

**命题 1.** 如果  $\llbracket \tilde{x} \vdash P; \tilde{n} \rrbracket = g$ ,  $\llbracket \tilde{x} \vdash Q; \tilde{n} \rrbracket = g'$ , 并且  $g$  和  $g'$  是同构的, 则  $\tilde{x} \vdash P \equiv Q; \tilde{n}$ .

**定理 1.**  $\tilde{x} \vdash P \equiv Q; \tilde{n}$  当且仅当  $\llbracket \tilde{x} \vdash P; \tilde{n} \rrbracket = \llbracket \tilde{x} \vdash Q; \tilde{n} \rrbracket$ .

证明. ( $\Rightarrow$ ) 由引理 1 可证.

( $\Leftarrow$ ) 由命题 1 可证.

由于在 Seal 演算中  $(\nu n)m[P]$  和  $m[(\nu n)P]$  并非结构同余 (其中  $n \neq m$ )<sup>[1]</sup>, 因此在  $'Seal$  中使用了类型标注机制以防止这两种进程表达式被转换成同构的偶图. 这两种表达式的偶图如图 3 所示.

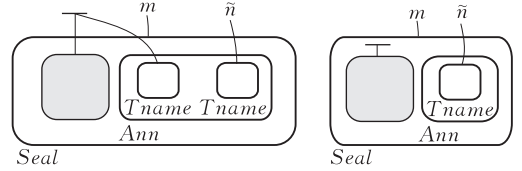


图 3 约束的位置

### 3.5 反应规则

在 BRS 理论中, 反应规则由  $R, R'$  和  $\eta$  组成<sup>[2]</sup>. 为了和 Seal 演算的反应规则相区别,  $'Seal$  反应规则的名字使用英文大写.

#### 'Seal 反应规则.

WRITE LOCAL:

$$R = Input_{x(u)} \oplus id_{n_0} \mid Output_{xv} \oplus id_{n_1}$$

$$R' = v/u \circ id_{n_0} \mid id_{n_1} \mid x/$$

$$\eta = \{0 \mapsto 0, 1 \mapsto 1\}$$

WRITE IN:

$$R = Output_{fn(x^y_1)u} \oplus id_{n_0} \mid (Seal_y \oplus id_{\tilde{m}}) \circ$$

$$(/z \circ (Input_{fn(x^y_2)u} \oplus id_{n_1} \mid id_{n_2}) \mid (Ann \oplus id_{\tilde{m}}) \circ \llbracket \tilde{m} \rrbracket)$$

$$R' = id_{n_0} \mid (Seal_y \oplus id_{\tilde{m}}) \circ (/z \circ (w/u \circ id_{n_1} \mid id_{n_2}) \mid$$

$$(Ann \oplus id_{\tilde{m}'} \circ \llbracket \tilde{m}' \rrbracket) \mid x/$$

$$\eta = \{0 \mapsto 0, 1 \mapsto 1, 2 \mapsto 2\}$$

WRITE OUT:

$$R = Input_{fn(x^y_1)u} \oplus id_{n_0} \mid (Seal_y \oplus id_{\tilde{m}}) \circ$$

$$(/z \circ (Output_{fn(x^y_2)v} \oplus id_{n_1} \mid id_{n_2}) \mid (Ann \oplus id_{\tilde{m}}) \circ \llbracket \tilde{m} \rrbracket)$$

$$R' = (/v \mid z) \circ (v/u \circ id_{n_0}) \mid (Seal_y \oplus id_{\tilde{m}}) \circ$$

$$(/z \setminus v) \circ (id_{n_1} \mid id_{n_2}) \mid (Ann \oplus id_{\tilde{m}'} \circ \llbracket \tilde{m}' \rrbracket) \mid x/$$

$$\eta = \{0 \mapsto 0, 1 \mapsto 1, 2 \mapsto 2\}$$

MOVE LOCAL:

$$R = Receive_{xu} \oplus id_{n_0} \mid Send_{xv} \oplus id_{n_1} \mid (Seal_v \oplus id_{\tilde{m}}) \circ$$

$$(id_{n_2} \mid (Ann \oplus id_{\tilde{m}}) \circ \llbracket \tilde{m} \rrbracket)$$

$$R' = id_{n_0} \mid (Seal_u \oplus id_{\tilde{m}}) \circ (id_{n_2} \mid (Ann \oplus id_{\tilde{m}}) \circ \llbracket \tilde{m} \rrbracket) \mid \cdots$$

$$\mid (Seal_{u_n} \oplus id_{\tilde{m}}) \circ (id_{n_2} \mid (Ann \oplus id_{\tilde{m}}) \circ \llbracket \tilde{m} \rrbracket) \mid$$

$$id_{n_1} \mid x/ \mid v/$$

$$\eta = \{0 \mapsto 0, 1 \mapsto 2, \dots, n \mapsto 2, n+1 \mapsto 1\}$$

MOVE IN:

$$R = Send_{fn(x^y_1)v} \oplus id_{n_0} \mid (Seal_v \oplus id_{\tilde{m}}) \circ (id_{n_1} \mid (Ann \oplus id_{\tilde{m}}) \circ$$

$$\llbracket \tilde{m} \rrbracket) \mid (Seal_y \oplus id_{\tilde{m}'} \circ (/z \circ (Receive_{fn(x^y_2)u} \oplus id_{n_2} \mid$$

$$id_{n_3}) \mid (Ann \oplus id_{\tilde{m}'} \circ \llbracket \tilde{m}' \rrbracket)$$

$$\begin{aligned}
R' &= id_{\bar{n}_0} | (Seal_y \oplus id_{\bar{m}''}) \circ (/z \circ (id_{\bar{n}_2} | id_{\bar{n}_3} | (Seal_{u_1} \oplus id_{\bar{m}}) \circ \\
&\quad (id_{\bar{n}_1} | (Ann \oplus id_{\bar{m}}) \circ \llbracket \tilde{m} \rrbracket) | \cdots | (Seal_{u_n} \oplus id_{\bar{m}}) \circ \\
&\quad (id_{\bar{n}_1} | (Ann \oplus id_{\bar{m}}) \circ \llbracket \tilde{m} \rrbracket) | (Ann \oplus id_{\bar{m}''}) \circ \llbracket \tilde{m}'' \rrbracket) | \\
&\quad x/ | v/ \\
\eta &= \{0 \mapsto 0, 1 \mapsto 2, 2 \mapsto 3, 3 \mapsto 1, \dots, n+2 \mapsto 1\} \\
\text{MOVE OUT:} \\
R &= Receive_{fn(x\eta)_u} \oplus id_{\bar{n}_0} | (Seal_y \oplus id_{\bar{m}}) \circ \\
&\quad (/z \circ (Send_{fn(x\eta)_v} \oplus id_{\bar{n}_1} | (Seal_v \oplus id_{\bar{m}''}) \circ \\
&\quad (id_{\bar{n}_2} | (Ann \oplus id_{\bar{m}''}) \circ \llbracket \tilde{m}' \rrbracket) | id_{\bar{n}_3} | (Ann \oplus id_{\bar{m}}) \circ \llbracket \tilde{m} \rrbracket) \\
R' &= id_{\bar{n}_0} | (/fn(R) | z) \circ ((Seal_{u_1} \oplus id_{\bar{m}'}) \circ (id_{\bar{n}_2} | (Ann \oplus \\
&\quad id_{\bar{m}''}) \circ \llbracket \tilde{m}' \rrbracket) | \cdots | (Seal_{u_n} \oplus id_{\bar{m}''}) \circ (id_{\bar{n}_2} | (Ann \oplus \\
&\quad id_{\bar{m}''}) \circ \llbracket \tilde{m}' \rrbracket) | (Seal_y \oplus id_{\bar{m}''}) \circ (/z \circ fn(R))) \circ \\
&\quad (id_{\bar{n}_1} | id_{\bar{n}_3} | (Ann \oplus id_{\bar{m}''}) \circ \llbracket \tilde{m}'' \rrbracket) | x/ | v/ \\
\eta &= \{0 \mapsto 0, 1 \mapsto 2, \dots, n \mapsto 2, n+1 \mapsto 1, n+2 \mapsto 3\}
\end{aligned}$$

### 3.6 操作相关

以 write local (WRITE LOCAL) 规则为例, 给出 Seal 演算反应规则与 'Seal 反应规则操作相关的证明方法。

**命题 2.** 在 write local 规则下  $\vdash p \searrow p' : \bar{n}$ , 当且仅当  $p$  和  $p'$  具有形式:

$$\begin{aligned}
\vdash p &\equiv \mathcal{E}(x^*(u).P \mid \bar{x}^*(v).Q) : \bar{n}, \\
\vdash p' &\equiv \mathcal{E}(P[v/u] \mid Q) : \bar{n},
\end{aligned}$$

其中  $\mathcal{E}$  是类型为  $\bar{n}$  的上下文。

**命题 3.** 在 WRITE LOCAL 规则下  $g \rightarrow g'$ , 当且仅当  $g$  和  $g'$  具有形式:

$$\begin{aligned}
g &= E \circ ((Input_{x(u)} \oplus id_{\bar{n}_0}) \circ h \mid (Output_{xv} \oplus id_{\bar{n}_1}) \circ h'), \\
g' &= E \circ ((v/u \circ id_{\bar{n}_0}) \circ h \mid id_{\bar{n}_1} \circ h' \mid x/),
\end{aligned}$$

其中  $E$  为活跃上下文,  $h$  和  $h'$  的外部接口分别是  $\bar{n}_0$  和  $\bar{n}_1$ 。

**引理 2.** 在 write local 规则下  $p \searrow p'$  当且仅当在 WRITE LOCAL 规则下  $g \rightarrow g'$ 。

证明. 根据命题 2,  $\vdash p \searrow p' : \bar{n}$ , 当且仅当  $p$  和  $p'$  具有形式

$$\begin{aligned}
\vdash p &\equiv \mathcal{E}(x^*(u).P \mid \bar{x}^*(v).Q) : \bar{n}, \\
\vdash p' &\equiv \mathcal{E}(P[v/u] \mid Q) : \bar{n},
\end{aligned}$$

根据结构同余和图同构可得到:

$$\begin{aligned}
\llbracket \vdash p : \bar{n} \rrbracket &= \llbracket \vdash \mathcal{E} : \bar{n} \rrbracket \circ \llbracket \vdash x^*(u).P \mid \bar{x}^*(v).Q : n' \rrbracket \\
&= \llbracket \vdash \mathcal{E} : \bar{n} \rrbracket \circ ((Input_{x(u)} \oplus id_{\bar{n}_0}) \circ \\
&\quad \llbracket \vdash P : \bar{n}_0 \rrbracket \mid (Output_{xv} \oplus id_{\bar{n}_1}) \circ \llbracket \vdash Q : \bar{n}_1 \rrbracket), \\
\llbracket \vdash p' : \bar{n} \rrbracket &= \llbracket \vdash \mathcal{E} : \bar{n} \rrbracket \circ \llbracket \vdash P[v/u] \mid Q : n' \rrbracket \\
&= \llbracket \vdash \mathcal{E} : \bar{n} \rrbracket \circ ((v/u \circ id_{\bar{n}_0}) \circ \\
&\quad \llbracket \vdash P : \bar{n}_0 \rrbracket \mid id_{\bar{n}_1} \circ \llbracket \vdash Q : \bar{n}_1 \rrbracket \mid x/).
\end{aligned}$$

令  $h = \llbracket \vdash P : \bar{n}_0 \rrbracket$ ,  $h' = \llbracket \vdash Q : \bar{n}_1 \rrbracket$ , 根据命题 3, 当且仅

当  $\llbracket \vdash p : \bar{n} \rrbracket \rightarrow \llbracket \vdash p' : \bar{n} \rrbracket$  时成立。

证毕。

使用类似方法可以分别证明 Seal 演算的其余 5 条反应规则与 'Seal 反应规则的对应, 限于篇幅, 证明略。

**引理 3.** 在 write in 规则下  $p \searrow p'$  当且仅当在 WRITE IN 规则下  $g \rightarrow g'$ 。

**引理 4.** 在 write out 规则下  $p \searrow p'$  当且仅当在 WRITE OUT 规则下  $g \rightarrow g'$ 。

**引理 5.** 在 move local 规则下  $p \searrow p'$  当且仅当在 MOVE LOCAL 规则下  $g \rightarrow g'$ 。

**引理 6.** 在 move in 规则下  $p \searrow p'$  当且仅当在 MOVE IN 规则下  $g \rightarrow g'$ 。

**引理 7.** 在 move out 规则下  $p \searrow p'$  当且仅当在 MOVE OUT 规则下  $g \rightarrow g'$ 。

**定理 2.** 对于每个类型良好的表达式  $\vdash p : \bar{n}$ ,  $\vdash p \searrow p' : \bar{n}$  当且仅当  $\llbracket \vdash p : \bar{n} \rrbracket \rightarrow \llbracket \vdash p' : \bar{n} \rrbracket$ 。

证明. 根据引理 2 到引理 7, 得证。证毕。

### 3.7 一个 Seal 演算反应的例子

Seal 演算下一个反应为

$$a[\bar{y}^b(\omega).R \mid b[y^*(z).S]] \searrow a[R \mid b[S[\omega/z]]],$$

$$\begin{aligned}
&(Seal_a \oplus id_{\bar{n}_0 \bar{n}_1 y b \omega}) \circ (Output_{y b \omega} \oplus id_{\bar{n}_0} | (Seal_b \oplus id_{\bar{n}_1 y}) \circ \\
&\quad (Input_{y(z)} \oplus id_{\bar{n}_1 z} | (Ann \oplus id_{\bar{n}_1 y}) \circ \llbracket \tilde{n}_1 y \rrbracket) | (Ann \oplus \\
&\quad id_{\bar{n}_0 \bar{n}_1 y b \omega}) \circ \llbracket \tilde{n}_0 \bar{n}_1 y b \omega \rrbracket) \rightarrow (Seal_a \oplus id_{\bar{n}_0 \bar{n}_1 b \omega}) \circ (id_{\bar{n}_0} | \\
&\quad (Seal_b \oplus id_{\bar{n}_1 \omega}) \circ (\omega/z \circ id_{\bar{n}_1 z} | (Ann \oplus id_{\bar{n}_1 \omega}) \circ \\
&\quad \llbracket \tilde{n}_1 \omega \rrbracket) | (Ann \oplus id_{\bar{m}}) \circ \llbracket m \rrbracket) | y/.
\end{aligned}$$

为了简单起见, 假定  $\bar{n}_0, \bar{n}_1, \{y\}, \{b\}, \{\omega\}, \{a\}$  和  $\{b\}$  互不相交. 对应的图形如图 4 所示。

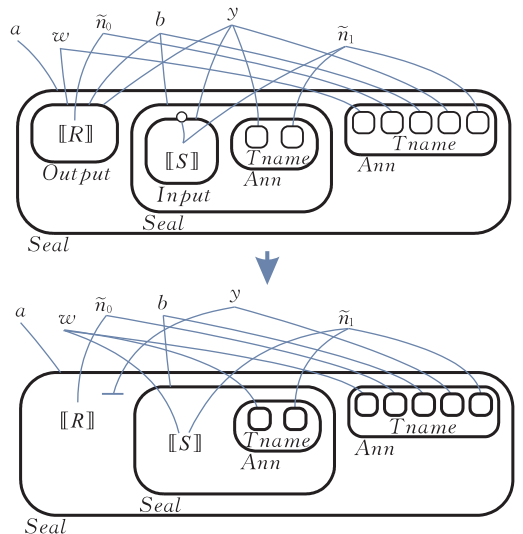


图 4 一个反应的例子

## 4 相关工作

在文献[2]中, Jensen 和 Milner 建立了偶图反应系统的基本理论, 展示了异步  $\pi$  演算  $A\pi$  的偶图表示, 并证明了从这种偶图表示中推导出的标签转换系统 (Labelled Transition System, LTS) 及其互相似性 (bisimilarity) 与  $A\pi$  传统的 LTS 和互相似性是匹配的. Milner 在文献[8]中给出了偶图静态结构 (即结构同余) 的公理系统. Milner 在文献[3]中给出条件-事件 Petri 网的偶图表示. Milner 还定义了绑定偶图和局部偶图理论<sup>[6]</sup>, 并用该理论给出了带有显式代换的一种  $\lambda$  演算变体的偶图表示. 在文献[5]中, 作者给出了一种嵌套位置、非线性活跃进程移动并带有局部名字的高阶进程演算——HOMER 的偶图表示.

偶图应用领域的研究也取得了许多成果. 丹麦哥本哈根 IT 大学正在开发基于偶图的普适计算语言——BPL (Bigraphical Programming Language)<sup>[9]</sup>, 围绕该语言, 一个综合了理论、技术、工具和方法学的通用框架正在逐步形成. BPL 语言很有可能代替现有的 UML 等建模语言, 成为普适计算时代主流的建模及开发语言.

## 5 结束语

偶图反应系统是一种新的理论工具, 其基础是一种强调位置和连接的移动计算图形化模型——偶图. 作为一种囊括现有并发和移动演算的元理论, 许多理论, 如 Petri 网、 $\pi$  演算、CCS、移动 Ambient、HOMER 和  $\lambda$  演算已经被描述成偶图反应系统. 本文扩展了文献[5]文中作者在 HOMER 偶图表示方面的研究成果, 给出了一种不带复制进程表达式的 Seal 演算的偶图表示, 分析了该 Seal 演算与其偶图表示间的结构对应和操作对应. 本研究扩展了偶图理论的应用范围, 展示了偶图理论在描述安全演算方面的能力. 作为构建 Seal 演算偶图反应系统的前期工作, 为在偶图框架下研究 Seal 演算的性质和应用奠定了基础.

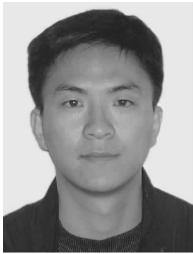
由于偶图反应系统理论还在完善过程中, 许多理论问题还没有彻底解决, 如对复制操作的偶图表示就还没有一种统一的认识. Milner 在<sup>[2]</sup>中讨论了复制 (replication) 和加和 (summation) 操作的表示

方法, 由于简单的表示方法会带来约束范围和变量复制等方面的一系列问题, 他没有在  $\pi$  演算的偶图表示中采用. 考虑到该问题的复杂性, 我们也没有在本文中将完整的 Seal 演算表示成偶图. 另外, 如何在偶图框架下分析 Seal 演算, 如推导出标签转换系统和互相似性, 并证明其与传统 Seal 演算中标签转换系统和互相似性的匹配将是下一步理论研究工作的重点. 而如何用 BPL 编码 Seal 演算, 将现有基于 Seal 演算的应用移植到 BPL 框架下, 将是未来应用研究工作的重点.

## 参 考 文 献

- [1] Castagna G, Vitek J, Nardelli F Z. The Seal calculus. *Information and Computation*, 2005, 201(1): 1-51
- [2] Jensen O H, Milner R. *Bigraphs and mobile processes (revised)*. Computer Laboratory, University of Cambridge, Cambridge, United Kingdom; Technical Report UCAM-CL-TR-580, 2004
- [3] Leifer J J, Milner R. *Transition systems, link graphs and Petri nets*. Computer Laboratory, University of Cambridge, Cambridge, United Kingdom; Technical Report UCAM-CL-TR-598, 2004
- [4] Milner R. *Pure bigraphs*. Computer Laboratory, University of Cambridge, Cambridge, United Kingdom; Technical Report UCAM-CL-TR-614, 2005
- [5] Bundgaard M, Hildebrandt T. *Bigraphical semantics of higher-order mobile embedded resources with local names*. IT University of Copenhagen, Copenhagen, Denmark; Technical Report TR-2005-70, 2005
- [6] Milner R. *Bigraphs whose names have multiple locality*. Computer Laboratory, University of Cambridge, Cambridge, United Kingdom; Technical Report UCAM-CL-TR-603, 2004
- [7] Birkedal L, Debois S, Elsborg E, Hildebrandt T T, Niss H. *Bigraphical models of context-aware systems*//Aceto L, Ingólfsdóttir A. *Proceedings of the 9th International Conference on Foundations of Software Science and Computation Structure*. Lecture Notes in Computer Science 3921, Springer-Verlag, 2006: 187-201
- [8] Milner R. *Axioms for bigraphical structure*. Computer Laboratory, University of Cambridge, Cambridge, United Kingdom; Technical Report UCAM-CL-TR-581, 2004
- [9] Birkedal L, Bundgaard M, Damgaard T C, Debois S, Elsborg E, Glenstrup A J, Hildebrandt T T, Milner R, Niss H. *Bigraphical programming languages for pervasive computing*//*Proceedings of the Pervasive 2006 International Workshop on Combining Theory and Systems Building in Pervasive Computing*. Dublin, Ireland, 2006: 653-658





**JIN Long-Fei**, born in 1977, Ph.D. candidate. His research interests include process calculi and semantic Web.

**LIU Lei**, born in 1960, professor, Ph. D. supervisor. His research interests include program analysis, process calculi, and semantic Web.

**Background**

This research is supported by the Research Fund for the Doctoral Program of Higher Education under grant No. 20061083044, and the Science Development Plan Project Found of Jilin Province under grant No. 20050527. The first project aims to bridge semantic Web domain and process calculi domain, and provide a methodology for researching semantic Web issues using process calculi theories. The second project aims to provide formal semantics of OMG' ontology definition meta-model (ODM) and apply these formal semantics in semantic Web application domain.

The theory of bigraphical reactive systems, due to Milner and co-workers, is based on a graphical model of mobile computation that emphasizes both locality and connectivity. A bigraph comprises a place graph, representing locations of computational nodes, and a link graph, representing interconnection of these nodes. Dynamics are given to bigraphs by defining reaction rules that rewrite bigraphs to bigraphs, roughly, a bigraphical reactive system (BRS) is a set of such rules. There are two principal aims for the new theo-

ry of bigraphical reactive systems: (1) to model ubiquitous systems; and (2) to be a meta-theory encompassing existing calculi for concurrency and mobility.

Along the second aim, the authors provide a bigraphical representation of the Seal calculus without replication in this paper, and analyze structural correspondence and operational correspondence between the Seal calculus and its bigraphical representation. All are foundations for researching characteristics and applications of Seal calculus within the bigraphical framework.

This research is the theory part of the two research projects above. Formal bridge between the logic foundation of semantic Web (such as description logics) and process calculi is built on it. The authors also pay their attentions on researching other important issues of semantic Web using theories of process calculi (especially bigraphs). These issues include ontology definition meta-model, ontology modeling, ontology analysis, ontology evolution and semantic Web applications, etc.