Reporte Técnico

Modelando el cálculo de Ambientes en *utcc*

John Alexander Vargas, Juan Francisco Díaz, Frank Valencia, Carlos Olarte

Octubre de 2009

1. Introducción

La programación concurrente por restricciones *ccp*, es un formalismo para modelar computacionalmente sistemas de agentes que interactúan entre ellos a través de un almacén de restricciones. *ntcc* es un cálculo basado en *ccp* para modelar sistemas reactivos adicionando el concepto de tiempo y no determinismo.

Un objeto (o proceso) se mueve cuando cambia sus conexiones de comunicación que puede ser visto como enlaces de interacción con otros procesos, o cuando cambia su localización o configuración espacial.

Esta noción de movilidad es una característica fundamental de algunos sistemas reactivos y *ntcc* y *ccp* han mostrado que modelar movilidad en ellos no es una tarea que se pueda realizar de forma natural. Esto es porque las variables lógicas solo se pueden asignar una vez y entonces, no se pueden enviar dos mensajes distintos por el mismo canal, pues de lo contrario, el almacen de restricciones se vuelve inconsistente. El cálculo de ambientes modela sistemas móviles a través de jerarquías y configuraciones espaciales de procesos, a partir de este cálculo, se definieron fórmulas de lógicas espaciales que describen propiedades de sistemas móviles como los modelados en π-cálculo y en ambientes.

El cálculo BioAmbients [Reg03] es una abstracción de sistemas biomoleculares que usa el π-cálculo para modelar los aspectos bioquímicos y moleculares, y el cálculo Ambientes, para modelar aspectos de localización molecular y compartimentos, incluyendo el movimiento de moléculas entre compartimentos.

De los cálculos que han surgido como extensiones del *ccp* se encuentra el *utcc* que permite modelar comportamiento infinito y movilidad proviendo un modelo que permite imitar el mecanismo de paso de nombres como se hace en π-cálculo [OlPalVal07]

La meta propuesta es explorar el uso del cálculo *utcc*, paramterizandolo con un sistema de restricciones basado en la lógica espacial del cálculo de ambientes, para modelar procesos con propiedades de localidad de sistemas móviles. En partícular se quiere modelar el sistema endocrino de regulación de peso corporal, que ha sido abordado como ejemplo para el cálculo bioambients. Esto involucra codificar propiedades de procesos del π-calculo y del cálculo ambients como fórmulas de la lógica espacial en el sistema de restricciones.

La metodología a seguir es:

1. Definir formalmente el sistema de restricciones basado en las lógicas espaciales definidas para el cálculo de ambientes.
2. Escoger un ejemplo sencillo modelado con el cálculo de ambientes y modelarlo con *utcc* y el sistema de restricciones definido.
3. Verificar que *utcc* parametrizado con el sistema de restricciones de lógicas espaciales satisface las propiedades descritas con esa misma lógica. Característica de *ccp* y sus extensiones.
4. Definir un modelo del sistema endocrino de regulación de peso corporal.
5. Verificar propiedades de localidad del modelo creado para el sistema endocrino de regulación de peso corporal.

Este reporte presenta una introducción del sistema de restricciones, un ejemplo sencillo modelado con *utcc* parametrizado con el sistema de restricciones e intentos de definir reglas de satisfacción de los procesos del cálculo para las fórmulas de la lógica.

1.1. Programación Concurrente por Restricciones

La programación concurrente por restricciones CCP es un modelo computacional donde cambia la idea original de variables de asignación y lectura como mecanismo de sincronización por un almacén de restricciones donde se imponen restricciones a través de un operador *tell* y se ejecutan procesos si ciertas restricciones se pueden deducir de las guardadas en el almacén de restricciones con un operador *ask*, [SarRP90]

$$\begin{array}{lll} P & ::= & D.A \\ D & ::= & \epsilon \,|\, p(X) :: A \,|\, D.D \\ A & ::= & c \,|\, c \rightarrow A \,|\, A \wedge A \,|\exists X.A \,|\, p(X) \end{array}$$

Sintaxis de CCP

Un sistema de restricciones simple es una estructura $< D, \vdash >$, donde D es un conjunto no vacio de tokens o restricciones primitivas. y $\vdash_{\subseteq} pD \times D$ es una relación de implicación (inferencia). pD es el conjunto de subconjuntos finitos de D que satisfacen:

- C1: $u \vdash P$ cada vez que $P \in u$
- C2: $u \vdash Q$ cada vez que $u \vdash P$ para todo $P \in v$ y $v \vdash Q$

Un aspecto importante a tener en cuenta es que para que un sistema de restricciones sea implementable la relación $\vdash$ debe ser decidible.

El almacen de restricciones en *ccp* es monotónico, es decir solo de pueden adicionar restricciones en él y estas no pueden modificarse o eliminarse.

$$\text{Si } < A_1, c > \rightarrow < A_2, d > \text{ es una transición interna posible y } e \geq c \text{ entonces}$$
$$< A_1, e > \rightarrow < A_2, d \sqcup e >$$

Monotocidad Operacional

Con la idea de definir un sistema de restricciones para modelar comportamiento de movilidad a través de propiedades espaciales, es necesario que el almacen de restricciones no permita comportamiento monotónico con el tiempo sobre la configuración espacial, es decir, poder modelar cuándo un proceso cambia su ubicación. De esta manera se debe intentar con un cálculo que permita estos cambios en el sistema de restricciones. *tcc* es un cálculo que es una extensión de *ccp* que adiciona la noción de tiempo, el cual es conceptualmente dividido en unidades discretas como intervalos de tiempo. El almacen de restricciones no se transfiere entre unidades de tiempo, a no ser que se especifice la imposición de una restricción como un proceso que replica en el tiempo. De esta manera se intentará definir el sistema de restricciones basado en la lógica espacial para este tipo de cálculo.

El cálculo *tcc* introduce operadores para esperar en una unidad de tiempo la ejecución de un proceso y para esperar si una condición se cumple en la unidad de tiempo actual; si no se cumple entonces un proceso será ejecutado en la siguiente unidad de tiempo.

$$\begin{array}{rcl} P, Q & ::= & \textbf{skip} \quad | \quad \textbf{tell}\ (c) \\ & & \textbf{when}\ c\ \textbf{do}\ P \quad | \quad P \parallel Q \\ & & (\textbf{local}\ x)\ P \quad | \quad \textbf{next}\ P \\ & & \textbf{unless}\ c\ \textbf{next}\ P \quad | \quad !P \end{array}$$

Sintaxis de tcc

El cálculo *utcc* remplaza el operador **when** c **do** P de *tcc* por (**abs** $\vec{x}$; c) P, estos procesos pueden ser vistos como λ-abstracciones del proceso P en las variables $\vec{x}$, sobre las restricciones c.

1.2. Cálculo de Ambientes

El cálculo de ambientes es uno de los formalismos nominales para modelar sistemas móviles. Aquí se ve la movilidad como la evolución de configuraciones espaciales sobre el tiempo [CarGor97]. En este cálculo los ambientes cumplen las siguientes características: Un ambiente puede estar dentro de otro ambiente. Un ambiente es un lugar limitado donde ocurre una computación. Un ambiente puede moverse como un todo. Cada ambiente tiene un nombre y una colección de subambientes. Los nombres son muy importantes, pueden ser creados o proporcionados para nombrar ambientes algunas veces de cuyas acciones pueden ser extraídas. A continuación se presenta la sintaxis de este cálculo de procesos.

$P,Q,R ::=$	processes	
$\mathbf{0}$	void	spatial
$P \mid Q$	composition	
$!P$	replication	
$M[P]$	ambient	
$M.P$	capability action	temporal
$(n).P$	input action	
$\langle M \rangle$	output action	
$M ::=$	messages	
n	name	names
$in\ M$	can enter into M	capabilities
$out\ M$	can exit out of M	
$open\ M$	can open M	
ε	null	paths
$M.M'$	composite	

Sintaxis del cálculo de Ambientes

Las operaciones de reducción que permiten expresar el comportamiento dinámico de ambientes en el sentido de poder entrar en un ambiente, poder salir de un ambiente o poder abrir el contenido del ambiente, son:

$n[in\ m.\ P \mid Q] \mid m[R] \longrightarrow m[n[P \mid Q] \mid R]$	(Red In)
$m[n[out\ m.\ P \mid Q] \mid R] \longrightarrow n[P \mid Q] \mid m[R]$	(Red Out)
$open\ n.\ P \mid n[Q] \longrightarrow P \mid Q$	(Red Open)
$(n).P \mid \langle M \rangle \longrightarrow P\{n \leftarrow M\}$	(Red Comm)
$P \longrightarrow Q \ \Rightarrow\ n[P] \longrightarrow n[Q]$	(Red Amb)
$P \longrightarrow Q \ \Rightarrow\ P \mid R \longrightarrow Q \mid R$	(Red Par)
$P' \equiv P, P \longrightarrow Q, Q \equiv Q' \ \Rightarrow\ P' \longrightarrow Q'$	(Red $\equiv$)
$\longrightarrow^*$ is the reflexive and transitive closure of $\longrightarrow$	

Reglas de Reducción del cálculo de Ambientes

En la primera regla de reducción se modela la acción de un ambiente con nombre n y con procesos P y Q en su interior, para entrar como un todo a otro ambiente llamado m. En la segunda relga se muestra la acción opuesta, salir de un ambiente. La tercera regla modela la capacidad para abrir el contenido de un ambiente y su interior queda expuesto. Se presenta la regla de comunicación. Por ejemplo el proceso $a[p[out\, a.in\, b. < m >]]|b[open\, p.(x).x[]]$ representa un proceso p que viaja saliendo del ambiente a, entrando al ambiente b, donde es abierto. Contiene un mensaje m que es leido y usado para crear un nuevo ambiente.

$a[p[out\ a.\ in\ b.\ \langle m\rangle]] \mid b[open\ p.\ (x).\ x[]]$	
$\rightarrow a[] \mid p[in\ b.\ \langle m\rangle] \mid b[open\ p.\ (x).\ x[]]$	(Red Out)
$\rightarrow a[] \mid b[p[\langle m\rangle] \mid open\ p.\ (x).\ x[]]$	(Red In)
$\rightarrow a[] \mid b[\langle m\rangle \mid (x).\ x[]]$	(Red Open)
$\rightarrow a[] \mid b[m[]]$	(Red Comm)

1.3. Lógicas Espaciales

Las lógicas espaciales[CarGor03] son un formalismo para especificar propiedades espaciales como jerarquía de localizaciones en el cálculo de ambientes. Como lógica modal, el valor de verdad depende del estado. En este caso depende del aquí (espacio) y el ahora (tiempo). Cada fórmula habla sobre el tiempo actual como estado actual de ejecución y lugar actual como localización actual. Por ejemplo la formula $n[0]$ es leida como "Aquí y ahora hay una lozalización vacia llamada n". También se han usado para describir el comportamiento y estructura espacial de sistemas concurrentes, en partícular del π-cálculo asíncrono[?]. La sintáxis es como sigue:

η, μ			Un nombre η o una variable x
A, B, C	$::=$	T	Verdad
		$\neg A$	Negación
		$A \vee B$	Disyunción
		0	Nulo
		$A \mid B$	Composición
		$A \rhd B$	Garantía
		$n[A]$	Locación
		$A@n$	Ubicación
		$n®A$	Revelación
		$A \oslash n$	Ocultamiento
		$n < n' >$	Mensaje
		$\Diamond A$	Modalidad en algún momento
		$\diamondsuit A$	Modalidad en algún lugar
		$\forall A$	Cuantificador Universal
		$И_{x.A}$	Cuantificador de Frescura

Las tres primeras fórmulas son como en las lógicas proposicionales, el valor de verdad, la disyunción y la negación. El nulo son propiedades que cumplen procesos nulos del cálculo, la composición indica la propiedad de dos procesos componen un proceso completo en una localización. La garantía $A \triangleright B$ es una propiedad satisfecha por todos los procesos que estando al lado (estar al lado quiere decir que componen un proceso) de un proceso que satisface A, se satisface o se deduce B. $n[A]$ es la propiedad que cumplen los procesos conformados por un lugar con nombre n y cuyo proceso interno satisface a A. Este operador es usado para ambientes. $A@n$ se cumple para los procesos que pueden ser ubicados en un lugar con nombre n. $n®A$ la revelación de n es una propiedad satisfecha por los procesos que tienen un nombre restringido n en procesos que satisfacen A. $A \oslash n$ es cumplida por los procesos en los cuales se puede restringir el nombre n. El operador $n < m >$ es definido para procesos con envio de mensajes a través de canales de comunicación del π-cálculo.

2. Modelando el cálculo de ambientes con *utcc* y un sistema de restricciones espaciales

2.1. Definición de un Sistema de Restricciones Espaciales (SRE).

Un sistema de restricciones representa información parcial de un sistema (Como un sistema de base de datos)[CarGio03]. Con el propósito de usar fórmulas de lógicas espaciales como restricciones básicas, describiendo la estructura de los datos en forma de árbol como lo hacen en [CrisCarGor02], se puede tener un sistema de restricciones donde su información sirva como modelo para la lógica espacial.

Se define el sistema de restricciones espaciales (SRE) $< D, \vdash >$, donde $D = D_S \otimes D_L$, D_S es un conjunto de estructuras de datos en forma de árbol que son descritas mediante fórmulas de la lógica espacial. D_L es un conjunto de predicados de lógica de primer orden para modelar los canales de comunicación. La relación$\vdash_{\otimes}$ queda definida como la composición de las reglas de inferencia y deducción definidas por $\vdash_{D_s}$ y $\vdash_{D_L}$.

Con las fórmulas de la lógica espacial se pueden especificar restricciones (propiedades) que deben cumplir las variables de dominio de grafos (árboles especificando configuración de locación), y con las de primer orden se pueden especificar propiedades de conecitividad.

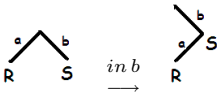
La idea es separar de los procesos del cálculo de ambientes, la información de como esta configurado espacialmente un ambiente y las acciones que se pueden ejecutar. La información sobre la configuración se impone en el sistema de restricciones mediante procesos **tell** de *utcc* y el operador de abstracción será

usado para modelar las acciones definidas en el cálculo de ambientes y el envio de mensajes al estilo del π-cálculo.

2.2. Modelando ambientes y acciones

- *Acción de entrar a un ambiente*

  $P \triangleq a[in\, b.R]$, $Q \triangleq b[S]$, entonces $P \mid Q \triangleq a[in\, b.R] \mid b[S] \rightarrow b[a[R] \mid S]$ que se ve representada en un esquema de árboles así:

  Un intento de modelar este comportamiento en *utcc* con el SRE podría ser, definiendo los siguientes procesos:

  $P \triangleq$ **tell** $(a[R])$

  $I \triangleq$ (**abs** $T_1, T_2; a[T_1] \mid b[T_2]$) **next tell** $(b[a[T_1] \mid T_2])$

  $Q \triangleq$ **tell** $(b[S])$

  entonces el proceso $P \,||\, Q \,||\, I$ iniciando con un almacen de restricciones vacio (con dominio de datos en estructura de árbol) tomaría las siguientes configuraciones:

  $< P \,||\, Q \,||\, I\,;\, \emptyset > \longrightarrow <\, Q \,||\, I\,;$  $> \longrightarrow <\, I;$  $> \Longrightarrow <\, skip\,;$  $>$

  El almacen de restricciones termina representando el comportamiento de la acción de entrar del cálculo de ambientes.

- *Acción de salir de un ambiente*

  $b[a[out\, b.R] \mid S] \rightarrow a[R] \mid b[S]$

  Intentando modelarlo en *utcc* con el SRE seri'a:

  $O \triangleq$ (**abs** $T_1, T_2; b[a[T_1] \mid T_2]$) **next tell** $(a[T_1] \mid b[T_2])$

  $P \triangleq$ **tell** $(b[a[R] \mid S])$ entonces $P \,||\, O$ iniciando con un almacen de restricciones vacio tomaría las siguientes configuraciones :

  $< P \,||\, O > \longrightarrow <\, Q;$  $> \Longrightarrow <\, skip\,;$  $>$

- *Acción de abrir un ambiente*

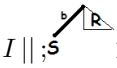
  $P \triangleq open\, b.R$, $Q \triangleq b[S]$ entonces $P|Q \rightarrow open\, R \mid b[S] \rightarrow R \mid S$

  Esto se puede modelar en *utcc* con el SRE así:

  $P \triangleq$ **tell** (R)

  $Op \triangleq$ (**abs** $T_1, T_2; T_1|b[T_2]$) **next tell** $(T_1|T_2)$

$Q \triangleq$ **tell** $(b[S])$ entonces $P \,||\, Q \,||\, Op$ iniciando con un almacen de restricciones vacio tomaría las siguientes configuraciones :

$< P \,||\, Q \,||\, Op\,;\, \emptyset > \longrightarrow < Q \,||\, I\,; R > \longrightarrow < I \,||\,;$  $> \Longrightarrow < skip\,;$  $>$

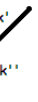
2.3. Ejemplo de Agente y el Firewall (Ambientes)

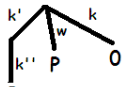
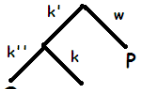
- $Firewall \triangleq (vw)w[k[out\, w.\, in\, k'.\, in\, w] \,|\, open\, k'.\, open\, k''.P]$
- $Agent \triangleq k'[open\, k.k''[Q]]$
- El proceso $(v\, k\, k'\, k'')(Agent \,|\, Firewall)$ reduce a $(v\, w)w[Q \,|\, P]$

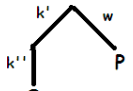
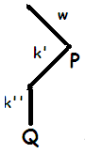
En *utcc* **con el SRE** Se definen los siguientes procesos:
$A \triangleq$ **tell** $(k'[k''[Q]])$
$B \triangleq$ (**abs** $T_1, T_2; k'[k[T_1] \,|\, T_2]$) **next tell** $(k'[T_1|T_2])$ /* Para modelar la acción $open\, k$ */
$C \triangleq$ (**local** w) **tell** $(w[P] \,|\, k[0])$
$D \triangleq$ (**abs** $T_1, T_2; k[T_1] \,|\, k'[T_2]$) (**next tell** $(k'[T_2|k[T_1]])$) || /* Para modelar la acción $in\, k'$ */
$E \triangleq$ (**abs** $T_3, T_4; k[T_3] \,|\, w[T_4]$) (**next tell** $(w[T_4|k[T_3]])$) || /* Para modelar la acción $in\, w$ */
$F \triangleq$ (**abs** $n, T_1, T_2; n[k'[T_1] \,|\, T_2]$) (**next tell** $(n[T_1|T_2])$) || /* Para modelar la acción $open\, k'$ */
$G \triangleq$ (**abs** $T_3, T_4; w[k''[T_3] \,|\, T_4]$) (**next tell** $(w[T_3|T_4])$) /* Para modelar la acción $open\, k''$ */
Entonces los procesos Agent y Firewall serian:

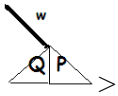
- $Agente \triangleq A \,||\, B$
- $Firewall \triangleq C \,||\, D \,||\, E \,||\, F \,||\, G$

Las configuraciones con un almacen de restricciones vacio para el proceso (**local** $k\, k'\, k'')(Agent \,||\, Firewall)$ sería:

$<$(**local** $kk'k'')(A \,||\, B \,||\, C \,||\, D \,||\, E \,||\, F \,||\, G);\, > \longrightarrow <$  $B \,||\, C \,||\, D \,||\, E \,||\, F \,||\, G;$ $> \longrightarrow$

$< B \,||\, D \,||\, E \,||\, F \,||\, G;$  $> \Longrightarrow <$  $B \,||\, E \,||\, F \,||\, G;$ $> \Longrightarrow$

$< E \,||\, F \,||\, G\,;$  $> \Longrightarrow <$  $F \,||\, G;$ $> \Longrightarrow <$  $G;$ $> \Longrightarrow < skip;$



3. Proponiendo *utcc* como módelo para la lógica espacial

Una de las características principales de los cálculos *ccp* es que los procesos satisfacen fórmulas de la lógica subyacente en el sistema de restricciones. Verificando la conservación de esa propiedad para *utcc* con el sistema de restricciones espaciales.

- **tell** $(c) \models c$, donde c es una fórmula de la lógica espacial.
- (**local** $x; c$) $P \vDash Hx.A$, para todo proceso $P \vDash A$
- (**abs** $x; c$) $P \vDash \mathcal{И} x.c \wedge Hx.A$, para todo proceso $P \vDash A$
- $\frac{P \models A \wedge Q \models B}{P \mid Q \models A \mid B}$
- **next** $P \models \circ A$, para todo proceso $P \vDash A$

El operador local hace la variable privada, es decir es oculta para los procesos. El operador abs usa nombres frescos para las variables que aparecen en el guarda, y son ocultas en el proceso abstraido. El paralelismo de dos procesos que imponen una estructuras de árbol, crean una composición de ellos.

4. Comentarios y trabajo en curso

- Es posible modelar sistemas de ambientes móviles en *utcc* parametrizado con una lógica espacial, como son modelados en el cálculo de ambientes y que el cálculo conserve la característica de *ccp* de modelar la lógica subyacente al sistema de restricciones.
- Para *utcc* modelar procesos como son modelados en π-cálculo, debe tener un sistema de predicados que indican los canales de envío de mensajes, mensajes que estan en las variables ligadas con el operador de abstracción.
- Para modelar procesos como en bioambients, creo que tocaría definir un sistema de restricciones con variables de dominio de grafos (árboles) y variables de dominio finito. *Un sistema de restricciones híbrido.*

Referencias

[CarGio03] Luca Cardelli, Giorgio Ghelli. *TQL: A Query Language for Semistructured Data Based on the Ambient Logic.* 2003

[CarGor97] Luca Cardelli y Andrew Gordón. *Mobile Ambients.* 1997.

[CarGor03] Luca Cardelli y Andrew Gordon. *Ambient Logic.* 2003

[CrisCarGor02] Cristiano Calcagno, Luca Cardelli, Andrew D. Gordon. *Deciding Validity in a Spatial Logic for Trees.* 2002

[OlPalVal07] Carlos Olarte, Catuscia Palamidesi y Frank Valencia. *Universal Timed Concurrent Constraint Programming.* 2007

[Reg03] Aviv Regev, E. Panina, W Silverman, L Cardelli y E. Shapiro. *BioAmbients: An abstraction for biological compartments.* 2003

[SarRP90] Vijay A Saraswat, Martin Rinard y Prakash Pamangaden. *Semantic foundations of concurrent constraint programming. 1990*