MODELAMIENTO DE SISTEMAS BIOLÓGICOS USANDO CÁLCULOS DE PROCESOS CONCURRENTES

Julian Gutiérrez, Jorge Andrés Pérez, Camilo Rueda

Grupo AVISPA
Pontificia Universidad Javeriana - Cali, Colombia
Calle 18 118-250, Cali, Colombia

E-mail: juliangutierrez@puj.edu.co , {japerez,crueda}@atlas.puj.edu.co

Los *cálculos de procesos* son formalismos propuestos para modelar sistemas concurrentes en diversos ámbitos. Su fundamentación matemática permite establecer *abstracciones* entre los elementos reales de los sistemas y los componentes básicos del cálculo, lo que facilita la verificación de propiedades interesantes de los sistemas modelados. Estas características perfilan a los cálculos de procesos como una alternativa interesante para especificar y verificar propiedades esenciales de sistemas biológicos. Este artículo presenta un análisis de los cálculos de procesos más importantes propuestos recientemente en el contexto biológico. Especial atención es dada a los cálculos *basados en restricciones*, que permiten, entre otras cosas, el manejo transparente de información parcial y cuantitativa.

Process calculi are formalisms that have been proposed to model concurrent systems in a wide variety of areas. Their mathematical foundations allow to define *abstractions* between the components of real systems and the constructions of a calculus. This is particularly convenient when verifying essential properties of the modelled systems. These features make process calculi an interesting approach to describe and verify biological systems. This article offers a survey of the process calculi proposed in the biological context. A particular class of calculi, those based on the notion of *constraint*, is thoroughly described. Those calculi provide, among other useful features, the transparent inclusion of quantitative and partial information into formal models.

Key Words: Process Calculi, Concurrent Constraint Programming, Molecular Biology.

Recibido: 30 agosto-2005 Revisado: 29 septiembre-2005 Aceptado: 18 diciembre-2005

0. INTRODUCCIÓN

Los cálculos de procesos son formalismos matemáticos diseñados para el estudio de sistemas de cómputo concurrentes. Pueden considerarse como lenguajes *abstractos* de programación, donde las ideas de *proceso* e *interacción* son centrales en la especificación de sistemas. Están definidos en términos de operaciones primitivas que establecen el tipo de interacciones posibles entre los procesos, así como su evolución en el tiempo. Este conjunto de operaciones promueve una construcción *composicional* de las especificaciones: la definición formal de un sistema tiene lugar por la composición de procesos simples que representan los subsistemas que lo conforman.

Estas características permiten considerar los cálculos de procesos como *instrumentos de abstracción* de sistemas dinámicos complejos. Esto significa que el modelamiento de un sistema utilizando cálculos de procesos se concentra en ciertos aspectos esenciales que determinan su comportamiento, mientras que otros, menos importantes de acuerdo con algún criterio particular, no son considerados. Este estilo de especificación, además, facilita enormemente la *verificación* de propiedades esenciales de los sistemas modelados.

Cada cálculo de procesos está concebido de forma que la especificación de sistemas es gobernada por algún criterio de abstracción particular. Por ejemplo, el calculo π [25] está basado en la idea de *nombres* como elemento fundamental para la descripción formal de sistemas de comunicación: ellos pueden ser simultáneamente tanto datos como canales de comunicación. Esquemas de comunicación de nombres permiten expresar formalmente la *movilidad* de los sitios que conforman un sistema distribuido. Bajo este contexto, en la especificación de un sistema de comunicación el diseñador debe concentrarse en la definición precisa de los sitios involucrados y de los medios por los cuales la comunicación tiene lugar (canales). Otras condiciones inherentes al sistema (e.g. ubicación geográfica de los sitios, condiciones técnicas de los canales, aspectos de seguridad) pueden considerarse como irrelevantes en el propósito de formalizar su comunicación, por lo que pueden ser ignorados en el modelamiento.

Es preciso anotar que el proceso de abstracción asociado con el modelamiento usando cálculos de procesos no afecta en lo absoluto la fidelidad de los modelos resultantes con respecto a las características del sistema real. Existen cálculos de procesos que admiten la parametrizacion de los modelos, por lo que por ejemplo, información cuantitativa obtenida de la experimentación puede considerarse de manera directa en el modelamiento. Adicionalmente, la composicionalidad de los cálculos favorece el refinamiento progresivo de los modelos con nuevos procesos que representen información adicional.

De esta manera, contando con abstracciones especificas que relacionen los elementos de los sistemas reales con los componentes de los cálculos de procesos, diversos

tipos de sistemas pueden ser formalizados. Algunos de estos sistemas son los estudiados en la *biología sistémica* (o *systems biology*). La biología sistémica puede entenderse como el estudio de los mecanismos presentes en los sistemas biológicos por medio de los cuales genes y proteínas se integran e interactúan dentro de un organismo [18]. Es decir, la biología sistémica estudia la estructura y expresión de un gen o un conjunto de genes. Este estudio, basado en la idea de sistema, presenta dos características fundamentales. Primero, se deben tener en cuenta las *interacciones* entre los componentes que conforman el sistema. Como consecuencia, se asume que los elementos que lo constituyen no se encuentran aislados del entorno sino que influyen y son influidos por él. Segundo, se debe poder analizar, modelar y simular, con mayor o menor detalle, el mismo sistema biológico. Esto implica poder identificar y abstraer sus propiedades fundamentales a distintos *niveles*.

La interesante similitud entre el estilo composicional de especificación propio de los cálculos de procesos y el propósito fundamental de la biología sistémica no ha pasado inadvertida entre investigadores en biología y las ciencias de la computación. Pueden identificarse varias etapas en esta relación de cooperación. Inicialmente, se propuso la abstracción de *comunicación como reacción* para modelar sistemas biológicos, utilizando cálculos de procesos propuestos originalmente para representar sistemas de comunicación. Tal acercamiento ha evolucionado en los últimos cinco años, en forma de cálculos de procesos diseñados específicamente para abordar el problema del modelamiento, simulación y verificación de propiedades de los sistemas biológicos. Cada uno de ellos se concentra en modelar elementos biológicos que constituyen abstracciones particulares. Es posible afirmar que en la actualidad nos encontramos en una etapa de transición, en donde ha sido identificada la necesidad de complementar los modelos teóricos (surgidos de los cálculos de procesos) con herramientas de software (simuladores, compiladores, entre otras) que los contextualicen en escenarios reales.

En este sentido, los cálculos de procesos *basados en restricciones* pueden contribuir contundentemente a la solución de problemáticas particulares en biología sistémica. En este tipo de cálculos, la abstracción de *restricción como información parcial* es la base de los modelos y lenguajes de programación. Los sistemas se definen por la información disponible, por lo que los modelos derivados son naturalmente extensibles. Los modelos basados en restricciones son independientes de los tipos de datos necesarios para representarlos, lo que facilita la inclusión de información cuantitativa proveniente de experimentos reales. Como ventaja adicional, los cálculos de procesos basados en restricciones poseen una agradable dualidad entre teoría y práctica, en forma de lenguajes de programación que permiten validar en la práctica modelos formulados en la teoría.

Este articulo constituye un sumario claro y conciso de las principales iniciativas de investigación en el área de cálculos de procesos concurrentes para la solución de problemas en biología sistémica. Su principal contribución consiste en presentar

de forma precisa la relación entre cálculos de procesos concurrentes y biología sistémica y molecular, con especial interés en los cálculos de procesos basados en restricciones.
El articulo se estructura de la siguiente manera. En la siguiente sección se presenta una introducción a los conceptos fundamentales de la biología molecular, mencionando algunos de los tipos de problemas de investigación en bioinformática. La sección 2. presenta un análisis de los principales cálculos de procesos que han sido propuestos para atacar problemas en biología sistémica, y de otros que se perfilan para hacerlo en los años venideros. Un especial énfasis es dado a los cálculos de procesos basados en restricciones. Finalmente, en la sección 3. se exponen algunas conclusiones.

1. CONCEPTOS EN BIOLOGÍA MOLECULAR Y PROBLEMAS EN BIOINFORMÁTICA

Para un mejor entendimiento del tipo de problemas que se han abordado hasta el momento con cálculos de procesos, se presenta a continuación una breve introducción a los conceptos fundamentales de la biología molecular y un sumario de algunos de los problemas en bioinformática. Estos conceptos e ideas son básicos para comprender el tipo de *abstracciones computacionales* hechas con los cálculos de procesos concurrentes a partir de hechos biológicos reales.

1.1. El dogma central

Los procesos moleculares dentro de la célula siguen un curso natural, una regla biológica denominada *el dogma central* de la biología molecular. Éste dogma establece que la información biológica no puede ser transferida de proteínas a proteínas ni de proteínas a ácidos nucléicos (ADN y ARN). Pocos elementos en la naturaleza no siguen este dogma. Uno de ellos son los *priones*, en los que es posible el flujo de información de proteínas a proteínas.
Así, la transferencia *estándar* de información biológica puede darse de tres formas distintas según el material inicial y final de cada proceso (ver Figura 1). Estas formas configuran los procesos de replicación, transcripción y traducción [23]. En la *replicación* la información biológica pasa del ADN al ADN, en la *transcripción* del ADN al ARN y en la *traducción* del ARN a proteínas.

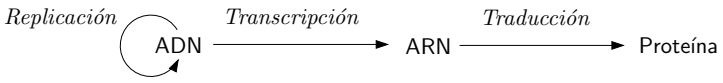

Figura 1. Dogma Central de la Biologia Molecular

Es necesario anotar que en algunos organismos puede existir flujo de información desde el ARN hacia el ADN. Esta transferencia de información es perfectamente posible según el dogma, ya que tiene lugar entre ácidos nucléicos. Sin embargo, esa ruta no es la más común en los organismos vivos.

1.1.1. Replicación

Este proceso consiste en generar dos cadenas de ADN hijas a partir de una cadena madre. La replicación de la información genética es fundamental en el proceso evolutivo ya que es una de las principales tareas en la división celular. Se dice que este proceso es *semi-conservativo* porque cada una de las cadenas nuevas conserva (o está constituida por) una de las hebras de la madre. La replicación, así como la transcripción y la traducción, presenta diferencias entre los organismos eucariotas y procariotas. Esto se debe, entre otras razones, al hecho de que los organismos eucariotas tienen membrana nuclear mientras que los procariotas no.

1.1.2. Transcripción

Este proceso consiste en copiar la información genética en forma de ADN en ARN. En organismos eucariotas, la transcripción incluye un proceso adicional (posterior a la síntesis de ARN) conocido como *splicing*. En el splicing se fragmenta la cadena de ARN inicial (conocida como ARN precursor) en subunidades llamadas *exones* e *intrones*. A partir de estas subunidades se genera una nueva cadena de ARN (llamada ARN mensajero) con algunos o todos los exones. Los intrones son eliminados completamente.

1.1.3. Traducción

Este proceso consiste en producir o sintetizar proteínas a partir de un molde de ARN. Las proteínas son cadenas de *aminoácidos*. En teoría, es posible la existencia de una gran variedad de aminoácidos distintos, pero solamente 20 tipos diferentes se utilizan para construir las proteínas [10]. El organelo encargado de hacer esta "traducción" de ácidos nucléicos a proteínas es el ribosoma.

1.2. Una Clasificación de Problemas Biológicos

Cuando se intenta entender (a nivel molecular) la composición, funcionamiento y evolución de los distintos componentes biológicos se encuentran varios problemas. Su naturaleza permite identificar al menos tres grandes grupos. Primero, problemas relacionados con *secuencias* de nucleótidos y aminoácidos. Segundo, problemas relacionados con la predicción de la *estructura* espacial de macromoléculas. Tercero, problemas relacionados con el *funcionamiento* de los sistemas. A continuación se describe cada uno de estos problemas y se hace referencia a trabajo previo en cada uno de ellos usando cálculos de procesos.

Problemas relacionados con secuencias

A pesar de que este tipo de problemas es uno de los más estudiados por la biología molecular computacional, es al mismo tiempo el menos tratado con cálculos de procesos. Esto se debe a que aquí lo que se busca es comparar, alinear o identificar nuevos patrones en secuencias de información biológica (ácidos nucléicos y proteínas). Así, se busca identificar por ejemplo, secuencias específicas en el ADN para el reconocimiento de genes o la presencia de motivos en proteínas. Por esta razón, la posición de los componentes de tales secuencias (y no su interacción) es el elemento fundamental en este tipo de problemas, haciendo inconveniente el uso de cálculos de procesos para su estudio.

Problemas relacionados con la estructura

Uno de los retos para los investigadores en biología y ciencias de la computación es poder definir la conformación espacial de un polímero (como el ADN, ARN o una proteína) a partir de la secuencia de los monómeros (nucleótidos o aminoácidos) que lo constituyen. Adicionalmente, un problema importante en este grupo consiste en la identificación de las leyes que gobiernan las interacciones entre dos o mas moléculas a través de sus *motivos*, lugares donde las moléculas pueden interactuar con otras. En los contextos teórico y práctico de los cálculos de procesos se ha utilizado la programación por restricciones [37] y el cálculo π [25] en [3, 30] y [33], respectivamente, para proponer soluciones aproximadas y hacer abstracciones de problemas de este tipo.

Problemas relacionados con la función

En la célula se pueden identificar tres tipos de tareas que definen todo su funcionamiento. Cada una de ellas es responsabilidad de un grupo de redes o rutas particulares. Estas son: las encargadas del *control celular*, las encargadas del transporte y transformación de sustancias *dentro de la célula* y las encargadas del transporte y transformación de sustancias *a través de la membrana celular*. Cada una se describe a continuación.

Redes de regulación de la expresión génica. Son las encargadas de todo el control celular. Este control puede estar a nivel de procesos o sistemas. Como esta regulación se fundamenta en la producción de *enzimas* que catalizan o inhiben los procesos celulares, la concentración de estas dentro de la célula resulta fundamental. Algunos modelos de estas redes están definidos en términos de sistemas de ecuaciones diferenciales que relacionan las concentraciones de los componentes involucrados en los procesos de control. La programación por restricciones ha sido utilizada para resolver problemas en esta área utilizando el calculo hcc [13].

Rutas metabólicas. Son el conjunto de todos los procesos en los que interactúan componentes biológicos para realizar el transporte y la transformación de sustancias en el *interior* de la célula. Por componente biológico se entiende toda conformación molecular y macromolecular como el agua, el oxígeno, los nutrientes

y los organelos intracelulares (por ejemplo las mitocondrias, los ribosomas y los cloroplastos) entre otros. Normalmente los procesos de una red metabólica son de tipo bioquímico. Estos procesos o *reacciones químicas* generalmente se expresan en términos de ecuaciones diferenciales. En el lenguaje Mozart [40] se han desarrollado algunas herramientas para encontrar rutas metabólicas con características restringidas [12].

Redes de transducción de señales. Son las encargadas de todos los procesos de interacción y transporte a través de la *membrana* celular. El transporte de sustancias puede ser activo o pasivo, dependiendo de si las sustancias se mueven a favor o en contra de gradientes electroquímicos existentes entre el interior y el exterior de la célula. Existen varios cálculos de procesos (ver sección 2.) que han sido especialmente desarrollados para modelar problemas de este tipo [6, 34].

2. CÁLCULOS DE PROCESOS CONCURRENTES

El principal atractivo de los cálculos de procesos concurrentes como herramienta de modelamiento de problemas de la vida real reside en el hecho de que son lenguajes de especificación expresivos, con estructuras matemáticas que permiten verificar o comprobar formalmente propiedades o condiciones de los sistemas que representan. Estos cálculos también se caracterizan por su generalidad, representada en diversos operadores que permiten modelar un amplio rango de situaciones. Esto ha permitido su aplicación en diversas áreas, desde la definición de sistemas distribuidos y móviles [11] hasta el control de robots [26], pasando por la informática musical [36], la logística y la bioinformática.

Dada la capacidad de representación de los cálculos de procesos concurrentes, resulta interesante preguntarse si estos formalismos son apropiados para el estudio riguroso de problemas en bioinformática. Esta área, por sus características e importancia en la vida cotidiana, se perfila como objeto de estudio interesante, con múltiples retos y desafíos por superar. En ese sentido, el modelamiento de problemas usando cálculos de procesos concurrentes puede constituirse en una metodología concreta para el descubrimiento y prueba de propiedades fundamentales en esta área.

En esta sección presentamos algunos de los trabajos que proponen el uso de cálculos de procesos para modelamiento de sistemas biológicos. Todos ellos tienen diversos orígenes y motivaciones: algunos se aprovechan de cálculos de procesos existentes (Cálculo π, CCS), otros proponen cálculos orientados específicamente al contexto biológico (Cálculo Brane, Biocalculus), mientras que algunos otras modifican cálculos existentes para abordar de forma más precisa ciertos elementos biológicos (Bioambients, CCS-R). Adicionalmente, esta sección presta especial interés a los cálculos de procesos basados en la noción de *restricción*. Este tipo de cálculos ofrece algunas ventajas interesantes para el contexto biológico, entre las

que se incluyen un manejo transparente de la información cuantitativa, además de la declaratividad y modularidad asociada con los modelos basados en restricciones.

2.1. Calculo π

Quizás el formalismo más representativo dentro de los cálculos de procesos es el calculo π [25], propuesto por Robin Milner a principios de los 90. Este calculo extiende CCS [24], su antecesor inmediato, con la idea de *nombres*, entidades que pueden ser tanto datos como canales de comunicación. Este principio, simple a primera vista, resulta poderoso para representar esquemas de comunicación complejos donde la topología del sistema puede cambiar.
La sintaxis del calculo π es la siguiente:

$$\left| \; P, Q \ldots \quad ::= \quad P \parallel Q \mid !P \mid (\nu x)P \mid \sum_{i \in I} \pi_i \cdot P_i \; \right|$$

en donde P y Q son procesos, x es un nombre y π es un *prefijo*. El prefijo denota una acción atómica previa a un proceso. Esta acción puede ser el envío o recepción de un dato por un canal. Por ejemplo, el envió de un dato z por el canal x se denota por $\overline{x}\langle z \rangle$; por su parte, la recepción de un dato b por el canal c se representa por $c(b)$. El proceso $P \parallel Q$ denota la composición paralela de los subprocesos P y Q. Además, permite que P y Q se comuniquen. El proceso $!P$ representa la *replicación* de P. Es decir, define la ejecución *infinita* de P. Un nombre puede estar restringido al contexto de un proceso particular. Este es el propósito de $(\nu x)P$, en donde el nombre x es local a P y solamente visible dentro de él. Finalmente, la escogencia no determinística entre una serie de procesos $\pi_i.P_i$ $(i \in I)$ se representa como su *sumatoria* $\sum_{i \in I} \pi_i.P_i$. Esta escogencia depende de la comunicación de dichos procesos. Dos procesos importantes pueden derivarse de la sumatoria: $\mathbf{0}$ y $\pi.P$. El primero representa la acción nula y es la abreviación de la sumatoria cuando $I = \emptyset$, mientras que el segundo se da cuando sólo hay un proceso involucrado en la escogencia.
El calculo π ha sido utilizado (tanto en su versión original como en una versión que lo extiende con elementos estocásticos [32]) por diversos autores para el modelamiento de sistemas en biología (ver por ejemplo [22, 8]). Sin embargo, las ideas más precisas al respecto provienen del grupo de trabajo liderado por Ehud Shapiro, quienes no sólo han usado el calculo para *representar* sistemas biológicos reales, sino que han trabajado intensamente en el desarrollo y uso de simuladores derivados de la extensión estocástica del calculo para el estudio de dichos sistemas. Su principal contribución consiste en una abstracción que asocia los elementos biológicos fundamentales con componentes del cálculo [33]. Dicha abstracción, denominada "*molécula como computación*" [35, 39], asocia de manera sencilla entidades y eventos presentes en el lenguaje con entidades y eventos biológicos. Por ejemplo, los motivos son asociados con los prefijos de entrada y salida del cálculo,

aprovechando la complementariedad que ambas entidades poseen en sus respectivos contextos.
En el campo aplicativo, la mayor contribución es BioPSI [2]. Se trata de una aplicación diseñada para la simulación de sistemas bioquímicos especificados en el cálculo π. Está construida sobre Flat Concurrent Prolog (FCP), lo que permite la implementación de las principales características del calculo π: movilidad y comunicación sincronizada. Existen dos compiladores disponibles para hacer simulaciones. El primero, denominado PI, permite especificar sistemas usando el cálculo original y ejecutar simulaciones semi-cuantitativas, mientras que SPI (el segundo compilador) utiliza una modificación de la versión estocástica del calculo, por lo que es apropiada para practicar simulaciones cuantitativas exactas. Tanto PI como SPI incluyen herramientas de depuración y de soporte para la simulación.
Para tener una idea más clara de la forma de un modelo en el cálculo π se presenta una descripción posible de la siguiente reacción química:

$$2\ H_2\ +\ O_2\ \stackrel{energia}{\Longrightarrow}\ 2\ H_2O$$

En el modelo se representa la validación de la existencia de los elementos antes de la reacción (hidrógeno y oxigeno). Se asume que de existir, los elementos están presentes en las cantidades apropiadas, es decir, dos moléculas de H_2 y una de O_2.

$$\begin{aligned} Hidrogeno &= \overline{h}\langle m\rangle.\mathbf{0} \\ Oxigeno &= \overline{o}\langle n\rangle.\mathbf{0} \\ Energia &= \overline{e}\langle p\rangle.\mathbf{0} \\ Reaccion &= e(i).h(j).o(k).agua \\ Sistema &= Hidrogeno \parallel Oxigeno \parallel Energia \parallel Reaccion \end{aligned}$$

El modelo anterior enfatiza el hecho de que la reacción es mediada por la presencia de energía en el sistema. El oxigeno, el hidrógeno y la energía se modelan como procesos independientes. También se modela la reacción en la que se produce agua a partir de los elementos ya mencionados. Finalmente, la interacción de los procesos se formaliza por medio del operador de composición paralela.
Como se ve, la presencia de los elementos involucrados en la reacción se modela como el envío de información por un canal asociado al proceso **0** en *Hidrogeno*, *Oxigeno* y *Energia*. En el proceso *Reaccion* se hacen las validaciones correspondientes a la reacción química modelada, es decir, si falta alguno de los componentes necesarios la reacción no tiene lugar. Es importante notar que éste tipo de modelos son una *abstracción* a muy alto nivel de procesos reales, por lo que otra información del sistema (como las cantidades de cada componente) no es tenida en cuenta.

2.2. CCS-R: modelando procesos reversibles

Una de las características inherentes a la mayoría de los cálculos de procesos es el incremento continuo de conocimiento durante la evolución de un proceso, o *monotonicidad*. De esta forma, los sistemas únicamente *progresan* durante la ejecución, sin posibilidad de retroceder o de deshacer alguna decisión tomada previamente. Esta condición, conveniente en el modelamiento y estudio de algunos sistemas (como por ejemplo, la verificación de protocolos de comunicación segura [9]), resulta opuesta a la esencia de muchos procesos bioquímicos, en donde "devolverse" a un estado anterior resulta natural, y en algunos casos, indispensable para describirlos apropiadamente.

Esta necesidad de modelar procesos *reversibles* fue la motivación principal de Danos y Krivine para desarrollar CCS-R [21], una extensión estricta del cálculo CCS [24]. Dicha extensión distingue las acciones posibles en el cálculo como *irreversibles* y *reversibles*. Las primeras se refieren a las acciones disponibles en el cálculo original. Los procesos relacionados con las segundas poseen un *identificador* y una *pila de memoria* que almacena elementos relativos a la comunicación entre procesos.

Para asegurar la coherencia en los retrocesos, CCS-R garantiza dos propiedades fundamentales. La primera, denominada *solidez*, asegura que por medio de retrocesos a etapas previas no se accedan estados previamente inalcanzables, mientras que la segunda (*expresividad*), garantiza que en las secuencias de acciones reversas no se dan como correctas dependencias causales falsas. La verdadera contribución detrás de CCS-R consiste en que estos dos principios, junto con los elementos de memorización mencionados antes, son incluidos explícitamente en las reglas formales que determinan la evolución de sistemas escritos en el cálculo.

Con respecto al modelamiento de sistemas biológicos, [21] identifica dos tipos de interacciones biológicas según la reversibilidad de sus acciones. Por un lado, en los mecanismos *elásticos* todos los componentes adoptan una configuración especial bajo la presencia de algún agente (o reactante) y retornan a la configuración original cuando tal agente no está presente. Por lo tanto, dichos mecanismos son susceptibles de ser modelados apropiadamente con las construcciones reversibles que provee el lenguaje. Por otro lado, los mecanismos *plásticos* representan aquellas interacciones en las que los cambios experimentados por sus componentes no pueden deshacerse. Como CCS-R *amplía* estrictamente CCS, este tipo de sistemas no reversibles pueden ser modelados también, sin perdida de generalidad, con CCS-R.

Para concluir, es posible afirmar que este trabajo provee intuiciones interesantes no sólo con respecto a la inclusión de elementos reversibles en los cálculos de procesos sino del rol de dichos elementos en sistemas biológicos. Existen dos preguntas válidas con respecto a este enfoque. Primero, desde una perspectiva técnica resulta interesante investigar si los mismos principios minimalistas presentes en el

diseño de CCS-R pueden ser aplicados a otro cálculo de procesos con mecanismos de comunicación más complejos, como el cálculo π o algún cálculo basado en el modelo de programación concurrente por restricciones. A priori, y basados en los principios fundamentales de dichos formalismos, la inclusión de reversibilidad en ellos no parece sencilla. Por otro lado, desde un punto de vista más fundamental, la existencia de mecanismos *totalmente* reversibles está cuestionada por el segundo principio de la termodinámica. La idea de reversibilidad total sin consumo de energía en CCS-R es sólo una idealización que dista de la realidad de los procesos físicos. De este modo, los análisis posibles a partir de modelos expresados en CCS-R estarán restringidos a idealizaciones con respecto a la energía de un sistema.

2.3. Brane: Un cálculo de interacción entre membranas

Uno de los principales problemas en bioinformática es el relacionado con los mecanismos por los cuales la célula se comunica con su exterior a través de la membrana. Tales mecanismos comúnmente involucran el traspaso de elementos (aminoácidos, oxígeno, agua, etc.) necesarios para el correcto desarrollo de los procesos celulares. Por lo tanto, alteraciones en este tipo de sistemas pueden descontrolar el funcionamiento de la célula, incluso al punto de llevarla a la muerte o a la generación de productos nocivos.

El calculo Brane [6] hace parte de los cálculos centrados en el modelamiento de las interacciones a través de membranas biológicas. La motivación es la abundancia de interacciones de éste tipo en los procesos celulares. En Brane la comunicación no se restringe a la realizada a través de la membrana citoplasmática sino que también incluye la realizada a través de la membrana nuclear y de otras como las membrana de los organelos (mitocondrias, cloroplastos, etc.) las cuales tienen una alta actividad dentro de la célula.

Dos aspectos fundamentales en el cálculo Brane son el tipo de *interacción* entre las membranas y el tipo de *transferencia* de información a través de ellas. Con respecto a lo primero, se pueden identificar tres tipos de interacción, a saber, la fagocitosis, la pinocitosis y la exocitosis. En ellos, la inspiración biológica está dada por el proceso de división celular y por los procesos en los que se acepta o se expulsa un agente biológico del interior de la célula. Por otro lado, la transferencia de la información a través de las membranas se puede clasificar en: paso de información por una membrana (por ejemplo comunicación a través de la membrana nuclear) y paso de información por dos membranas (por ejemplo paso de información del citoplasma de una célula a otra) [31].

Por otro lado, en el campo aplicativo es poco lo que se ha logrado. Pese a que con Brane, como en todos los cálculos de procesos analizados en este artículo, es posible verificar propiedades de los sistemas modelados, no existe aún una herramienta que

implemente las ideas formales del cálculo. Mientras no se avance en esa dirección no es posible verificar la utilidad real que pueda tener el cálculo a nivel biológico.

2.4. BioAmbients: espacios de computación biológica

Este calculo también está enfocado en los problemas relacionados con las interacciones a través de membranas. BioAmbients [34] proviene del cálculo Ambient [7], por lo que conserva sus principales características como la creación de ambientes independientes y móviles en los que se ejecutan procesos del cálculo π.
BioAmbients define tres tipos de comunicación y seis primitivas para la manipulación de los ambientes. Los tipos de comunicación son, primero, uno local entre procesos de un mismo ambiente (dos procesos en paralelo), segundo, uno entre procesos localizados en ambientes que se encuentren a un mismo nivel jerárquico (dos ambientes en paralelo) y, tercero, uno entre procesos localizados en distintos ambientes en los que uno de ellos tiene un nivel jerárquico mayor que el otro (un ambiente se encuentra dentro de otro). Por otro lado, se tienen primitivas para *entrar*, *aceptar*, *expulsar*, *salir* y *mezclar* ambientes [31].
Una ventaja de este cálculo es que permite pensar en un sistema como un conjunto de partes que interactúan, pero que a su vez cada parte (o componente biológico en este caso) tiene unas características que lo definen y que se encuentran dentro de un ambiente independiente. Otra ventaja es la facilidad de incluir un modelo dentro de otro simplemente encerrándolo dentro de un ambiente y definiéndole una interfaz de interacción con el exterior. Una desventaja es la necesidad de complementariedad entre los componentes biológicos para su interacción, algo que realmente no es siempre necesario a menos que se trate de una interacción a través de motivos moleculares.
Desde lo práctico BioAmbients cuenta con un herramienta de simulación llamada BioPSI, descrita anteriormente en la sección 2.1..

2.5. Bio-Calculus: interacciones moleculares en biología sistémica

Este calculo, cuya sintaxis fue presentada en [27], intenta suplir la necesidad de expresar de forma clara y fácil muchas de las interacciones posibles entre moléculas. Bio-calculus pretende cumplir con 4 requerimientos básicos de expresión:

- Componentes biológicos inmersos en un sistema más complejo.
- Transición y comunicación.
- La inclusión de información para procesos de simulación.
- Sentencias con un fundamento matemático que posibiliten el desarrollo de demostraciones formales.

La satisfacción de estos requerimientos genera un puente entre el lenguaje usado por investigadores en la biología y en las ciencias de la computación.
Una característica especial de este cálculo es la relación entre una sintaxis inspirada en la biología y un sistema *multi-semántico*. Esto significa que, a diferencia de los

lineamientos generales de los cálculos en donde existe una única semántica que define la manera en que los sistemas evolucionan o se transforman, Bio-calculus cuenta con un sistema que le permite evolucionar de diferentes formas a partir de un mismo modelo, dependiendo de la semántica que se escoja. Algunas de estas semánticas son:

- Una *estocástica* y *continua* en la que todas las moléculas son tratadas de forma uniforme.
- Una en la que las reacciones entre moléculas están definidas por una *ecuación diferencial* que representa la reacción bioquímica.
- Una en la que el comportamiento del sistema evoluciona de acuerdo a la ecuación de *Michaelis-Menten*.
- Una en la que cada molécula es tratada *individualmente* y en la que se escogen *estocásticamente* pares de moléculas para que interactúen.

Finalmente, vale la pena mencionar que este cálculo permite modelar sistemas con distintos niveles de detalle [19, 20], desde moléculas tratadas individualmente hasta grupos de ellas modeladas simplemente por su concentración dentro de una reacción, la cual puede a su vez, obedecer al comportamiento determinado por una ecuación diferencial o por una distribución de probabilidad.

2.6. Cálculos de Procesos Basados en Restricciones

El modelo de programación concurrente por restricciones (cc) [37] propone el concepto de *restricción* como elemento de *información parcial*, en contraposición al concepto convencional de *variable* utilizado en los lenguajes de programación imperativos. Las restricciones se acumulan en un *almacén* que contiene toda la información del sistema; dicho almacén es *monotónico* en cuanto la información conocida únicamente crece con el tiempo. Los agentes concurrentes (procesos) que componen el sistema tienen acceso sobre el almacén, el cual es el medio único de sincronización y comunicación entre ellos.

En general, el modelo cc ofrece operaciones para adicionar una restricción al almacén (**tell** c), consultar la presencia de una restricción en el almacén (**ask** c), ocultar información en un proceso (**local** X **in** P) y ejecutar dos procesos en paralelo ($P \parallel Q$). La sintaxis exacta varía de acuerdo a cada cálculo.

En los últimos años se han propuesto varios cálculos basados en restricciones [15, 37, 38, 16, 28, 36]. Un aspecto particularmente interesante de todos ellos consiste en su vasta aplicabilidad en el modelamiento de problemas de la vida real. Esto puede apreciarse en múltiples trabajos de tipo aplicativo, que van desde extensiones, paquetes y librerías para el manejo de restricciones hasta la creación de lenguajes de programación basados en el modelo cc [40], pasando por simuladores y herramientas de soporte para el modelamiento de sistemas [41].

A continuación se presentan detalladamente `ntcc` y hcc, dos de los cálculos de procesos basados en restricciones propuestos recientemente. Más adelante, se re-

sume una serie de ventajas que tienen este tipo de cálculos sobre otros enfoques similares.

2.6.1. Calculo hcc (Hybrid Concurrent Constraint)

hcc [16] es un cálculo de procesos basado en restricciones que permite modelar tiempo continuo y discreto. Para entender sus características, es preciso describir antes su evolución así como la relación con otros cálculos del modelo cc.

Como se mencionó antes, una de las principales características del modelo cc es que permite detectar la *presencia* de información, por medio de la operación primitiva **if** c **then** P, que ejecuta el proceso P si c se deduce de la información presente en el almacén de restricciones. Sin embargo, ciertos sistemas pueden describirse de forma más adecuada por medio de condicionales que evalúen la *ausencia* de información. En dichos casos, la operación descrita es inapropiada para el modelamiento, porque en el caso en que c no se deduzca del almacén, la ejecución del sistema se bloquea indefinidamente.

El problema consiste entonces en detectar que una restricción no se deduce del almacén. La primera solución consistió en la inclusión del concepto de *fases de ejecución*, induciendo así una noción de temporalidad. De esta forma, la ausencia de información se evalúa al *final* de una fase de ejecución. Bajo este contexto, se adicionaron dos operaciones al modelo. La primera, **if** c **else next** P, ejecuta el proceso P en el *siguiente* instante de tiempo si no se deduce c en el actual. La segunda, **hence** P, ejecuta P en todos los instantes de tiempo después del actual. Este modelo extendido se conoce como tcc (Timed Concurrent Constraint Programming) [38].

A pesar de esta extensión, en ciertos sistemas es indispensable reaccionar *inmediatamente* cuando cierta información no está disponible. Para superar esta falencia se propuso Default cc, en donde se *estima* un almacén final antes de iniciar la computación. Más específicamente, Default cc agrega la operación **if** c **else** P, que ejecuta P *por defecto* y de manera *inmediata* si no de deduce c del almacén estimado. Al final del cómputo, el almacén obtenido y el estimado inicialmente son comparados. Si son iguales, la estimación fue correcta y se acepta el cómputo. En caso contrario se produce un fallo.

Finalmente, hcc extiende Default cc adicionándole la noción de fases de ejecución sobre tiempo continuo. En otras palabras, un programa hcc ejecuta múltiples programas Default cc, uno en cada fase de ejecución. Cuando en alguna de las fases de ejecución un programa Default cc falla (almacén estimado y obtenido distintos) se elimina esa rama de los cómputos o se hace backtracking. La sintaxis precisa de hcc se presenta en el cuadro 1.

Tabla 1

Sintaxis de hcc

Primitiva	**Proposición**
c	Impone la restricción c
if c **then** P	Ejecuta el proceso P si c se deduce del almacén
new X **in** P	Crea la variable X local en P
P , Q	Ejecuta los procesos P y Q en paralelo
if c **else** P	Ejecuta el proceso P si c no se deduce del almacén
hence P	Ejecuta P siempre al principio de cada fase

2.6.2. Cálculo `ntcc`*: Tiempo y No determinismo*

`ntcc` es un cálculo de procesos concurrentes que considera el concepto de restricción en un contexto donde el *tiempo* es definido y manipulado explícitamente. Esto permite, entre otras cosas, el modelamiento preciso de comportamiento no determinístico y asíncrono [28]. A continuación se enumeran brevemente las principales características de `ntcc`, proporcionando detalles de su sintaxis.

Tiempo. En `ntcc` el tiempo se divide conceptualmente en *unidades* o *intervalos*. En una unidad de tiempo, un proceso puede recibir un estímulo proveniente del ambiente que tiene influencia en el proceso actual y origina un conjunto de información que es el punto de partida para el próximo intervalo de tiempo. Cada intervalo de tiempo tiene asociado su propio almacén de restricciones; la información disponible en un almacén de restricciones dado no se transfiere automáticamente al siguiente.

Comunicación. La comunicación en `ntcc` sigue las ideas del modelo de programación por restricciones: es posible añadir y obtener información parcial sobre las variables de un sistema en términos de operaciones *ask* y *tell*. Más precisamente, el proceso **when** c **do** P consulta el almacén de restricciones actual sobre la presencia de c. Si durante el intervalo de tiempo en que se pregunta c puede inferirse del almacén, se ejecuta P dentro de esa unidad de tiempo. De otro modo, este proceso es excluido de toda ejecución futura.

Alternativas de Ejecución. Dado que la información parcial permite representar diferentes valores para las variables de un sistema, resulta conveniente poder modelar diversas escogencias y alternativas de acción. En `ntcc` es posible expresar escogencias *no determinísticas* extendiendo el proceso **when** c **do** P de la siguiente manera:

$$\sum_{i \in I} \textbf{when } c_i \textbf{ do } P_i$$

Esta expresión representa un proceso que debe escoger no deterministicamente, en el intervalo actual de tiempo, un proceso P_j ($j \in i$) cuya guarda c_j se deduzca del almacén. Esta escogencia es excluyente.

Dependencias Temporales. Este es quizás el aspecto más atractivo de `ntcc`, pues el cálculo provee construcciones que permiten representar dependencias temporales entre procesos. Es decir, se toma una determinada acción dependiendo de la ausencia o presencia previa de restricciones. Existen dos operadores de este tipo:

- **next** P, que representa la activación del proceso P en el siguiente intervalo de tiempo.
- **unless** c **next** P, que activa P en la siguiente unidad de tiempo siempre y cuando no se pueda deducir c del almacén actual de restricciones.

Comportamiento Asíncrono e Infinito. Con el operador $\star$ es posible extender de manera arbitraria (pero finita) la espera o retardo en la ejecución de un proceso. De este modo, es posible modelar *comportamiento asíncrono* en procesos. Así por ejemplo, $\star$ **tell**(c) representa la eventual adición de la restricción c, sin información precisa con respecto al instante de tiempo en que esto ocurrirá. Por otra parte, el operador ! define la ejecución infinita de un proceso dado, ejecutando una copia del proceso en cuestión por infinitas unidades de tiempo futuras. Por ejemplo, el proceso $P =$! **tell**(temperatura > 0) especifica que durante toda la ejecución del sistema, el valor de la temperatura debe ser positivo.

Para ilustrar la noción de *información parcial* subyacente a los cálculos de procesos basados en restricciones considérese el siguiente ejemplo. Sean dos partículas P y Q que habitan en un espacio bidimensional. De esta forma, cuatro variables (denotadas P_x, P_y, Q_x y Q_y) representarán la posición horizontal y vertical de ambas partículas. Por simplicidad, asumiremos que los valores posibles para estas variables son los números naturales. Si bien esta situación es muy general para considerarse un ejemplo biológico en si mismo, puede considerarse como el fundamento de problemas biológicos relevantes en la actualidad, como la predicción de la estructura de las proteínas y el ARN.

Suponga que en una primera instancia la información disponible del sistema se resume en las desigualdades $P_x \leq 4$, $P_y > 0$, $Q_x \leq 2$ y $1 \leq Q_y \leq 2$. Esto se representa gráficamente como en la parte superior izquierda de la Figura 2 (las posiciones del espacio posibles para P y Q se denotan con $\bigcirc$ y $\triangle$, respectivamente).

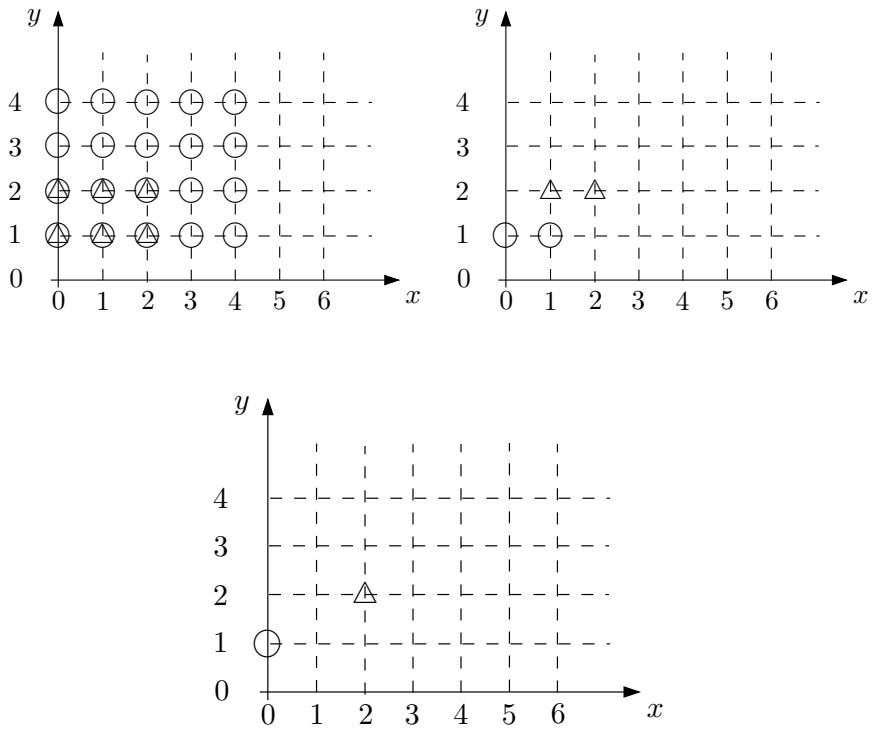

Figura 2. Posición inicial, intermedia y final para las partículas P y Q.

Asuma que en un momento posterior la información del sistema se incrementa con $P_y < Q_y$ y $Q_x > P_x$. Esto reduce las posibilidades de posicionamiento para P y Q, como se aprecia en la parte superior derecha de la Figura 2. Finalmente, si consideramos que la distancia existente entre P y Q debe ser mayor que 2 (es decir, $\sqrt{(P_x - Q_x)^2 + (P_y - Q_y)^2} > 2$), entonces se logra determinar de forma única la posición para P y Q, como se ilustra en la parte inferior de la Figura 2.
La información de la posición de P y Q puede formalizarse de la siguiente manera utilizando `ntcc`. Cada F_i representa un estado del conocimiento del sistema; estos estados pueden integrarse para lograr una versión más completa de todo el sistema.

$$
\begin{aligned}
F_1 &= \mathbf{tell}(P_x \leq 4) \parallel \mathbf{tell}(Q_x \leq 2) \parallel \mathbf{tell}(Q_y \leq 1) \parallel \mathbf{tell}(P_y > 0) \\
F_2 &= \mathbf{tell}(P_y < Q_y) \parallel \mathbf{tell}(Q_x > P_x) \\
F_3 &= \mathbf{tell}(\sqrt{(P_x - Q_x)^2 + (P_y - Q_y)^2} > 2) \\
Sistema &= F_1 \parallel F_2 \parallel F_3
\end{aligned}
$$

Este ejemplo permite apreciar como modelos iniciales con poco nivel de detalle pueden hacerse más precisos de forma progresiva, en forma de restricciones que representan nuevo conocimiento. En este ejemplo, otros detalles podrían ser mod-

elados de forma inmediata, como la información sobre la posición en Z de ambas partículas. De la misma forma, información cuantitativa adicional (en términos de números reales, por ejemplo) podría mejorar la descripción del modelo propuesto. En términos de descripción dinámica, los operadores temporales disponibles en `ntcc` podrían servir para modelar el movimiento de P y Q. Por ejemplo, el movimiento vertical de P (desde el origen) podría especificarse de la siguiente manera:

$$\mathbf{tell}(P_x = 0.0) \parallel \mathbf{tell}(P_y = 0.0) \parallel \mathbf{tell}(P_x = 0.0) \parallel \mathbf{tell}(P_y = 0.25).$$

2.6.3. Ventajas del modelo de programación por restricciones en la biología sistémica

La descripción de sistemas biológicos usando cálculos de procesos basados en restricciones [4] involucra una serie de características que pueden resultar benéficas para los intereses de la biología. Algunas de ellas son:

- Contar con *agentes* concurrentes permite una clara separación entre diferentes procesos biológicos. Además, existe la posibilidad de mejorar el modelo mediante el *refinamiento* progresivo de los procesos.
- La *modularidad* de los modelos permite aislar su núcleo y conservar así ciertas propiedades fundamentales, aún cuando el entorno cambie constantemente. Esto facilita la integración entre modelos.
- El estilo *declarativo* de la programación permite que únicamente las restricciones tienen que ser definidas en el modelo. El control del programa es formalmente definido por el cálculo.
- El uso de *restricciones* permite expresar el conocimiento de una manera parcial o incompleta. Dependiendo del número y detalle de éstas se define la granularidad del análisis del modelo, es decir, se puede trabajar sobre distintos *niveles* del mismo sistema. Este ultimo punto determina una clara ventaja del modelo de restricciones para el estudio de problemas en biología sistémica sobre otros enfoques.

Adicionalmente, la generalidad que el modelo de programación por restricciones otorga al concepto de *restricción* permite la coexistencia de varios *sistemas de restricciones* que definan reglas sobre distintos tipos de datos. Por ejemplo, restricciones sobre enteros, números reales y conjuntos pueden utilizarse en el mismo modelo. Esto no sólo facilita el manejo de información cuantitativa sino que también constituye una ventaja para el modelamiento, pues los problemas pueden abordarse usando un acercamiento discreto, continuo o estocástico [5, 14] sin mayores inconvenientes.

Algunos trabajos interesantes usando el modelo de restricciones son el modelamiento de un subproceso de la regulación de la transcripción y algunas interacciones entre genes con hcc [13], la búsqueda de redes bioquímicas utilizando

Mozart [12], el modelamiento de un sistema biológico usando procesos estocásticos no determinísticos [29], la predicción de la estructura espacial de una cadena de aminoácidos usando modelos *lattice* [3] y *off-lattice* [30]. Sin embargo, las últimas investigaciones apuntan a nuevos descubrimientos en systems biology con el uso de restricciones [4]. Existen trabajos de investigación que exploran las ventajas y desventajas de usar hcc [13] y `ntcc` [17] bajo el enfoque de la biología sistémica.

Uno de estos trabajos ([17]) estudia un mecanismo celular de transporte activo conocido como la *bomba de sodio-potasio*. Dicha bomba procura el mantenimiento de gradientes electrostáticos entre el interior y el exterior de la célula, con el propósito de gestionar el ingreso adecuado de nutrientes. Los modelos en `ntcc` de este mecanismo resultan más expresivos que modelos desarrollados en otros cálculos (en particular [8]), no sólo por la inclusión de tiempo sino por la facilidad de incluir información cuantitativa por medio de restricciones. Esto es, magnitudes como la concentración de sustancias pueden ser modeladas de forma intuitiva usando un sistema de restricciones apropiado. Además de lo anterior, el modelamiento está soportado por el sistema de prueba de propiedades asociado con el cálculo. Esto permite que propiedades interesantes del sistema sean verificadas con medios (semi)automáticos como probadores de teoremas o simuladores específicos.

3. CONCLUSIONES

Los cálculos de procesos se presentan como una alternativa viable para modelar y probar propiedades de sistemas biológicos. En particular, las características de los cálculos los hacen propicios para modelar y resolver problemas relacionados con la estructura y la función de sistemas biológicos complejos.

Esta posibilidad de aplicación ha sido reconocida oportunamente por la comunidad científica interesada en concurrencia y cálculos de procesos. De los trabajos analizados pueden identificarse dos vertientes principales de investigación. La primera se vale de los resultados teóricos existentes para el modelamiento de sistemas biológicos. De esta forma, la investigación gira alrededor del modelamiento de los sistemas más que del estudio y/o extensión de las propiedades formales del cálculo. El trabajo de Shapiro [39] es una clara muestra de esta dirección. La segunda vertiente se concentra en la descripción de problemáticas biológicas tan específicas que requieren un cálculo de procesos especializado. Por esta razón, la novedad de las propuestas radica en la alta especialización de las primitivas de los cálculos, así como de los fundamentos semánticos asociados. Los cálculos inspirados en membranas [6, 34] son un buen ejemplo de esta situación. Si bien ambas vertientes persiguen en mayor o menor medida la aplicación de los resultados, son las propuestas basadas en cálculos existentes las que han logrado resultados más concretos en términos biológicos, por medio de la construcción de simuladores y

otras herramientas de software. Estos avances sin embargo, no ocultan la urgente necesidad de un mayor esfuerzo en la producción de software orientado a los expertos en biología y química, siendo éste desafío el reto más importante en esta área.

En ese sentido, los cálculos basados en restricciones gozan de unas características muy atractivas. Además de las ya mencionadas ventajas derivadas de los fundamentos del modelo de programación por restricciones, en este tipo de cálculos pueden distinguirse dos áreas complementarias de investigación. La primera de ellas, que podría denominarse *teórica*, tiene que ver con el estudio y desarrollo de los componentes formales de los cálculos de procesos, como la sintaxis y su semántica asociada. En esta área se enfatiza en la necesidad de desarrollar modelos biológicos completos, que sean coherentes dentro del contexto formal del cálculo. La investigación por lo tanto, debe concentrarse en la extensión de las capacidades de los cálculos de modo que el desarrollo de modelos compactos y expresivos esté formalmente sustentado. La inclusión de parámetros cuantitativos en modelos formales es tan solo un ejemplo de este tipo de extensiones. La segunda área de estudio es más *practica* y comprende el desarrollo y mantenimiento de lenguajes de programación y librerías de software basadas en restricciones. El objetivo aquí es incorporar técnicas del estado del arte de la programación por restricciones y la computación a los componentes de software existentes, de modo que el rendimiento de los programas se incremente. La implementación de técnicas que faciliten la descripción de los modelos dentro de las aplicaciones es también una dirección abierta. El ejemplo más representativo de un lenguaje de programación basado en el modelo de programación concurrente por restricciones es Mozart [40]; una librería genérica de restricciones que implementa los principales lineamientos del modelo que ha sido recientemente propuesta es GECODE [1]. Ambas áreas, teórica y práctica, además de complementarse y orientarse entre si, tienen perspectivas prácticas claras. Mientras que por la investigación básica en la primera es posible vislumbrar la construcción de simuladores y herramientas similares, las aplicaciones de software que pueden construirse por la segunda pueden tener alcances prácticos considerables.

Por estas razones, el trabajo futuro en los cálculos basados en restricciones se cierne promisorio. En el campo aplicativo, la inclusión de estructuras y/o mecanismos que induzcan la noción de tiempo parece ser el paso inicial. En este contexto, el desarrollo de sistemas de restricciones que incorporen ecuaciones diferenciales es una alternativa interesante por su alta aplicabilidad. Paralelamente, el mantenimiento y fortalecimiento de los sistemas de restricciones existentes sobre números reales y dominios finitos debe ser ejecutado. En el campo teórico, la formalización de algunos conceptos relacionados con el *comportamiento* biológico resulta muy interesante. Por ejemplo, el estudio del significado de *equivalencia* en procesos biológicos puede conducir a resultados muy diferentes a las técnicas clásicas desarrolladas en

los últimos años. El estudio formal de otros conceptos relacionados, como las nociones de *distancia* entre componentes biológicos y de comportamiento de un proceso en un *contexto* particular son también interesantes. No menos importante resulta la inclusión de elementos probabilísticos en ambos ámbitos. Otra dirección de trabajo consiste en la definición, diseño e implementación de máquinas abstractas *correctas* que reflejen fielmente los conceptos formalizados por los cálculos, al tiempo que ofrezcan características esenciales del software, en particular eficiencia y extensibilidad.

REFERENCIAS

1. GECODE. Generic Constraint Development Environment. Available at www.gecode.org.
2. The BIOPSI project, 2005. Available at http://www.wisdom.weizmann.ac.il/ biopsi/.
3. Rolf Backofen. Using constraint programming for lattice protein folding. In *German Conference on Bioinformatics*, page 123. Oxford University Press, 1997.
4. Alexander Bockmayr and Arnaud Courtois. Using hybrid concurrent constraint programming to model dynamic biological systems. In Peter J. Stuckey, editor, *ICLP*, volume 2401 of *Lecture Notes in Computer Science*, pages 85–99. Springer, 2002.
5. J. M. Bower and H. Bolouri, editors. *Computational modeling of genetic and biochemical networks*. MIT Press, 2001.
6. Luca Cardelli. Brane calculi. In Vincent Danos and Vincent Schachter, editors, *CMSB*, volume 3082 of *Lecture Notes in Computer Science*, pages 257–278. Springer, 2004.
7. Luca Cardelli and Andrew D. Gordon. Mobile ambients. In Maurice Nivat, editor, *FoSSaCS*, volume 1378 of *Lecture Notes in Computer Science*, pages 140–155. Springer, 1998.
8. G. Ciobanu, V. Ciubotariu, and B.Tanasa. A pi-calculus model of the Na pump. In *Genome Informatics 2002*, pages 469–472. Universal Academy Press.
9. F. Crazzolara and G. Winskel. Events in security protocols. In Pierangela Samarati, editor, *Proceedings of the 8th ACM Conference on Computer and Communications Security*, pages 96–105, Philadelphia, PA, USA, November 2001. ACM Press.
10. Helena Curtis and N. Sue Barnes. *Biologia*, chapter Moleculas Organicas, page 85. Editorial Medica Panamericana, 2000.
11. Juan Francisco Díaz and Camilo Rueda. Modelos para la computación móvil (invited paper). *Revista Colombiana de Computación*, 1(1):29–45, 2000.
12. Grégoire Dooms, Yves Deville, and Pierre Dupont. A mozart implementation of cp(bionet). In Peter Van Roy, editor, *MOZ*, volume 3389 of *Lecture Notes in Computer Science*, pages 237–250. Springer, 2004.
13. Damien Eveillard, Delphine Ropers, Hidde de Jong, Christiane Branlant, and Alexander Bockmayr. A multi-scale constraint programming model of alternative splicing regulation. *Theor. Comput. Sci.*, 325(1):3–24, 2004.
14. M. A. Gibson and E. Mjolsness. *Computational modeling of genetic and biochemical networks*, chapter Modeling the activity of single genes. MIT Press, 2001.
15. Vineet Gupta, Radha Jagadeesan, and Vijay A. Saraswat. Models for concurrent constraint programming. In Ugo Montanari and Vladimiro Sassone, editors, *CONCUR*, volume 1119 of *Lecture Notes in Computer Science*, pages 66–83. Springer, 1996.

16. Vineet Gupta, Radha Jagadeesan, Vijay A. Saraswat, and Daniel G. Bobrow. Programming in hybrid constraint languages. In *Hybrid Systems*, pages 226–251, 1994.

17. Julian Gutierrez, Jorge A. Perez, Camilo Rueda, and Frank D. Valencia. Time, Nondeterminism and Constraints in Modeling and Verifying Biological Systems. In preparation., 2006.

18. BioSeek Inc. Systems biology-what is it?, 2005.

19. H. Kitano, editor. *Foundations of Systems Biology*. MIT Press, 2001.

20. H. Kitano. *Foundations of Systems Biology*, chapter Systems Biology: Toward System-level Understanding of Biological Systems. MIT Press, 2001.

21. J. Krivine and V. Danos. Formal molecular biology done in CCS-R. In *BioConcur 2003, Workshop on Concurrent Models in Molecular Biology*, 2003.

22. Celine Kuttler and Joachim Niehren. Gene regulation in the pi calculus: Simulating cooperativity at the lambda switch. In *BioConcur: Workshop on Concurrent Models in Molecular Biology*, volume Extended Version, August 2004. Extended Version.

23. Benjamin Lewin. *Genes VII*. Oxford University Press, 2000.

24. R. Milner. *Comunication and Concurrency*. International Series in Computer Science. Prentice Hall, 1989.

25. R. Milner. *Communicating and Mobile Systems: The π-Calculus*. Cambridge University Press, 1999.

26. Pilar Muñoz and Andrés René Hurtado. Programming robotic devices with a timed concurrent constraint language. In Mark Wallace, editor, *CP*, volume 3258 of *Lecture Notes in Computer Science*, page 803. Springer, 2004.

27. Masao Nagasaki, Shuichi Onami, Satoru Miyano, and Hiroaki Kitano. Bio-calculus: Its concept and molecular interaction. *Genome Informatics*, 10:133–143, 1999.

28. Mogens Nielsen, Catuscia Palamidessi, and Frank Valencia. Temporal concurrent constraint programming: Denotation, logic and applications. *Nordic Journal of Computing*, 9:145–188, 2002.

29. Carlos Olarte and Camilo Rueda. Using stochastic ntcc to model biological systems. In *Proceedings of CLEI2005*, 2005.

30. Alessandro Dal Palù, Agostino Dovier, and Federico Fogolari. Protein folding simulation in ccp. In Bart Demoen and Vladimir Lifschitz, editors, *ICLP*, volume 3132 of *Lecture Notes in Computer Science*, pages 452–453. Springer, 2004.

31. D. Prandi, C. Priami, and P. Quaglia. Process calculi in a biological context. *Bulletin of the EATCS*, February 2005.

32. Corrado Priami. Stochastic π-calculus. *The Computer Journal*, 38(6):578–589, 1995.

33. A. Regev and E. Shapiro. *Modelling in Molecular Biology*, chapter The π-calculus as an abstraction for biomolecular systems, pages 219–266. Natural Computing Series. Springer, 2004.

34. Aviv Regev, Ekaterina M. Panina, William Silverman, Luca Cardelli, and Ehud Y. Shapiro. Bioambients: an abstraction for biological compartments. *Theor. Comput. Sci.*, 325(1):141–167, 2004.

35. Aviv Regev and Ehud Shapiro. Cells as Computation. *Nature*, 419:343, September 2002.

36. Camilo Rueda, Gloria Alvarez, Luis O. Quesada, Gabriel Tamura, Frank D. Valencia, Juan Francisco Díaz, and Gerard Assayag. Integrating constraints and concurrent objects in musical applications: A calculus and its visual language. *Constraints*, 6(1):21–52, 2001.

37. V. Saraswat, M. Rinard, and P. Panangaden. The semantic foundations of concurrent constraint programming. In *POPL '91*, pages 333–352, Jan 1991.

38. Vijay Saraswat, Radha Jagadeesan, and Vineet Gupta. Foundations of timed concurrent constraint programming. In *Proceedings, Ninth Annual IEEE Symposium on Logic in Computer Science, Paris, France, 4–7 July, 1994*, pages 71–80, 1109 Spring Street, Suite 300, Silver Spring, MD 20910, USA, 1994. IEEE.

39. Ehud Y. Shapiro. Molecule as computation: Towards an abstraction of biomolecular systems. In Roderic Guigó and Dan Gusfield, editors, *WABI*, volume 2452 of *Lecture Notes in Computer Science*, page 418. Springer, 2002.

40. G. Smolka. The Oz programming model. In Jan van Leeuwen, editor, *Computer Science Today*, volume 1000 of *LNCS*, pages 324–343. Springer - Verlag, 1995.

41. V. Saraswat and R. Jagadeesan and V. Gupta. jcc: Integrating timed default concurrent constraint programming into Java. In *Proceedings of EPIA 2003*, volume 2902 of *LNCS*, 2003.