APLICACIONES REALES DE MODELOS BIOINSPIRADOS

Tema 4: Modelización computacional de sistemas dinámicos complejos: Un marco bioinspirado.

David Orellana Martín
Mario de J. Pérez Jiménez

Grupo de investigación en Computación Natural
Dpto. Ciencias de la Computación e Inteligencia Artificial
Universidad de Sevilla

Máster Universitario en Lógica, Computación e Inteligencia Artificial

Curso 2024-2025

Objetivos

- Metodología para diseñar un marco de modelización en **Membrane Computing**.
- Marco formal de especificación basado en sistemas P.
- Sistemas P multientornos:
  - ⋆ Orientación estocástica.
  - ⋆ Orientación probabilística.
- Sistemas P multicompartimentales.
  - ⋆ Algoritmo multicompartimental de Gillespie.
  - ⋆ Algoritmo determinista de tiempo de espera.
- Sistemas PDP (Population Dynamics P systems).
  - ⋆ Algoritmo BBB.
  - ⋆ Algoritmo DNDP.
  - ⋆ Algoritmo DCBA.
- Sistemas PGP (Probabilistic Guarded P systems).
  - ⋆ Algoritmo de simulación.

Paradigma de la computación celular con membranas

Membrane Computing: **Gh. Păun**, 1998–2000.

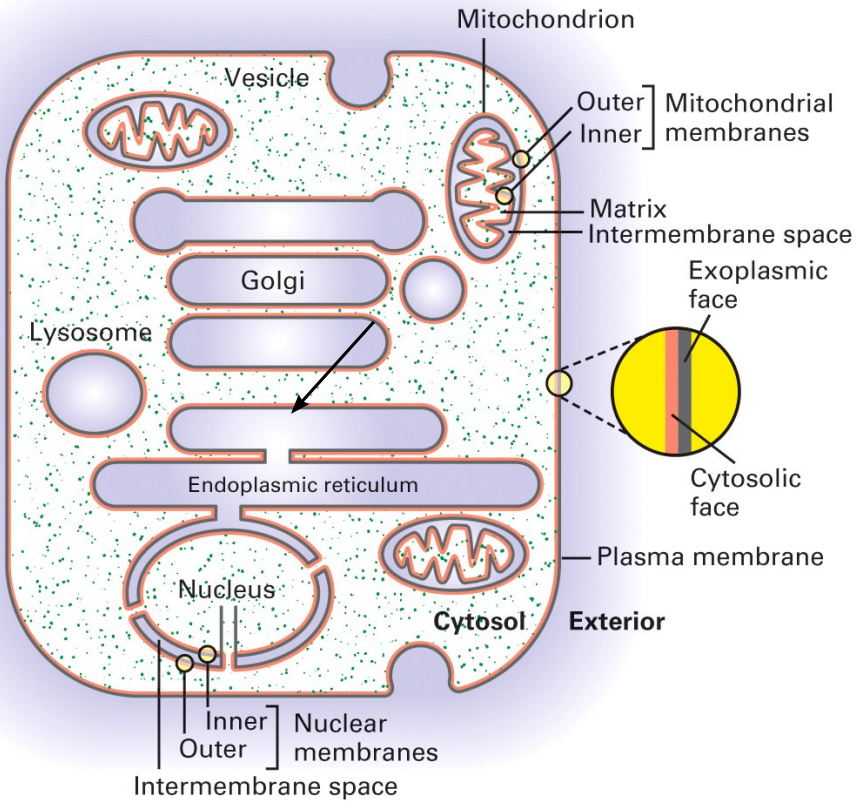
- Modelos de computación (**sistemas de membranas** o **sistemas P**):
  - Orientados a máquinas.
  - No deterministas.
  - Distribuidos.
  - Paralelos y maximales.
- Los modelos de computación celular pueden trabajar:
  - A modo de células (**cell-like**).
  - A modo de tejidos (**tissue-like**).
  - A modo de neuronas (**neural-like**).

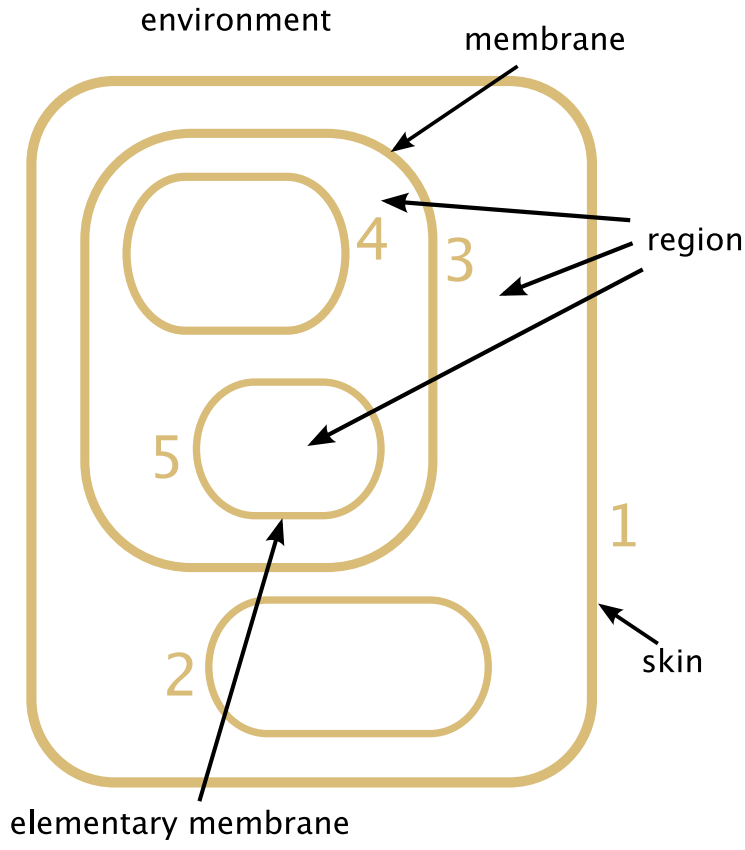
Sistema básico de membranas que trabaja a modo de células

Ingredientes **sintácticos**:

- **Alfabeto de trabajo**(los elementos se denominan **objetos**).
- Un subconjunto del alfabeto de trabajo (objetos de **entrada**).
- Una **estructura de unidades de procesos** (membranas): **árbol enraizado**.
- Un **multiconjunto** asociado a cada unidad de proceso.
- Un conjunto finito de **reglas de evolución**.
- Un **entorno pasivo**: **sólo recibe objetos**.
- Una membrana distinguida (de **entrada**).

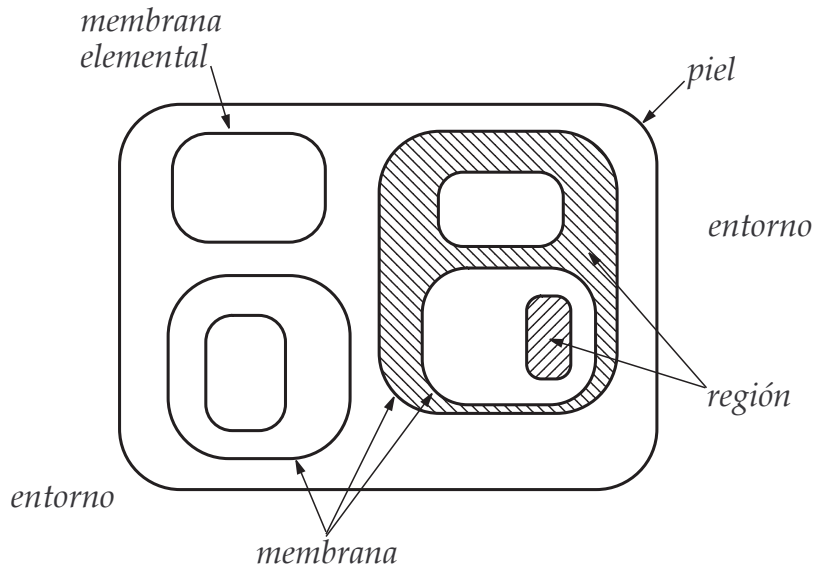
Sistema básico de membranas que trabaja a modo de células





Sistema básico de membranas que trabaja a modo de células

Una **estructura de membranas** (diagrama de Venn):



Formalmente, una estructura de membranas es un **árbol enraizado etiquetado**:

- ⋆ La raíz del árbol es la membrana piel.
- ⋆ Las hojas son las membranas elementales.
- ⋆ En este contexto, el "padre" de la membrana piel es el "entorno" del sistema.

Las membranas pueden tener etiquetas y polarizaciones ($\{0, +, -\}$).

Sistema básico de membranas que trabaja a modo de células

Sistema de membranas que trabaja a modo de células, de grado $q \geq 1$:

$$\Pi = (\Gamma, \Gamma', \mu, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}, i_{in})$$

en donde:

* Γ es un alfabeto finito (de **trabajo**) y $\Gamma' \subsetneq \Gamma$ (Γ': alfabeto de **entrada**).
* μ es una **estructura de membranas** de grado q: membranas etiquetadas biyectivamente con elementos de $\{1, \ldots, q\}$ y polarizaciones de $\{0, +, -\}$.
* Para cada i, $1 \leq i \leq q$, $\mathcal{M}_i$ es un multiconjunto finito sobre Γ.
* $\mathcal{R}$ es un conjunto de **reglas de evolución** asociadas a las membranas. Son del tipo $u\,[\,v\,]_i^{\alpha} \longrightarrow u'\,[\,v'\,]_i^{\alpha'}$ siendo $u, v, u', v' \in M(\Gamma)$, $1 \leq i \leq q$, $\alpha, \alpha' \in \{0, +, -\}$
* $i_{in} \in \{1, \ldots, q\}$ representa la **membrana de entrada** del sistema.

Sistema básico de membranas que trabaja a modo de células

Configuración de Π en un instante t: una tupla $C_t = (\mathcal{M}'_1, \ldots, \mathcal{M}'_q)$ tal que $\mathcal{M}'_i$ es el multiconjunto de objetos sobre Γ contenido en la membrana i, en ese instante.

Configuración inicial de Π: $\boxed{(\mathcal{M}_1, \ldots, \mathcal{M}_q)}$

Configuración inicial de Π asociada a un multiconjunto m sobre Σ:

$$\boxed{(\mathcal{M}_1, \cdots, \mathcal{M}_{i_{in}} + m, \ldots \mathcal{M}_q)}$$

Aplicabilidad de una regla $u\,[\,v\,]_i^{\alpha} \longrightarrow u'\,[\,v'\,]_i^{\alpha'}$ a una configuración C_t.

- ⋆ En la estructura de membranas de C_t ha de aparecer la membrana etiquetada por i con carga eléctrica α.
- ⋆ El multiconjunto v ha de estar contenido en esa membrana i y el multiconjunto u ha de estar contenido en su padre.

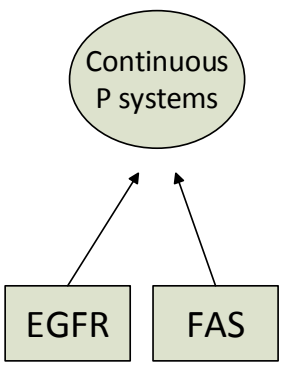
Multiconjunto de reglas aplicables a una configuración.

Sistema básico de membranas que trabaja a modo de células

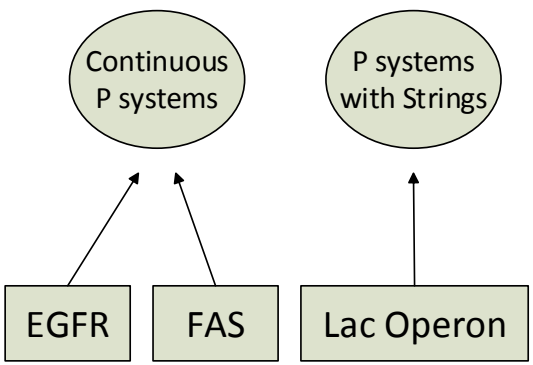
Sean $C' = (\mathcal{M}'_1, \ldots, \mathcal{M}'_q)$ y $C'' = (\mathcal{M}''_1, \ldots, \mathcal{M}''_q)$ configuraciones de Π:

- $\star$ C'' se obtiene de C' en **un paso de transición** ejecutando las reglas aplicables de $\mathcal{R}$ (de forma **paralela y maximal**) como sigue:
  - Si $u\,[\,v\,]_i^{\alpha} \longrightarrow u'\,[\,v'\,]_i^{\alpha'}$ es una regla aplicable asociada a la membrana i, entonces
    - $\star$ El multiconjunto v se elimina de esa membrana y el multiconjunto u se elimina de su padre.
    - $\star$ Los objetos del multiconjunto v' se añaden a la membrana i y los objetos del multiconjunto u' se añaden al padre de i.
    - $\star$ La membrana i pasará a tener carga α'.
  - **Paralelismo**: En cada membrana, todas las reglas serán aplicadas en paralelo. En el sistema, todas las membranas trabajan simultáneamente.
  - **Maximalidad**: tras la ejecución de las reglas no puede quedar un objeto por evolucionar al que se le pueda aplicar alguna regla.

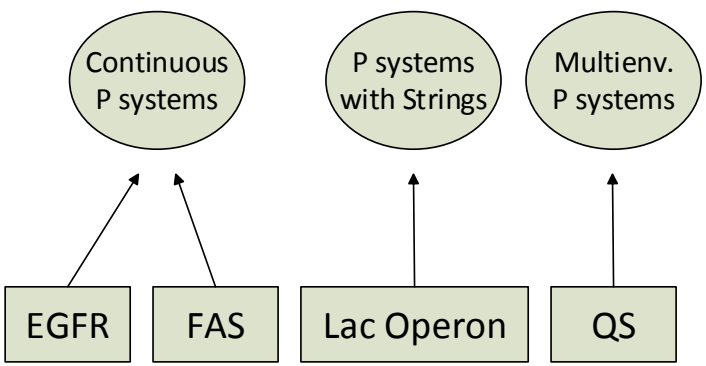
Metodología para diseñar un marco de modelización en Membrane Computing



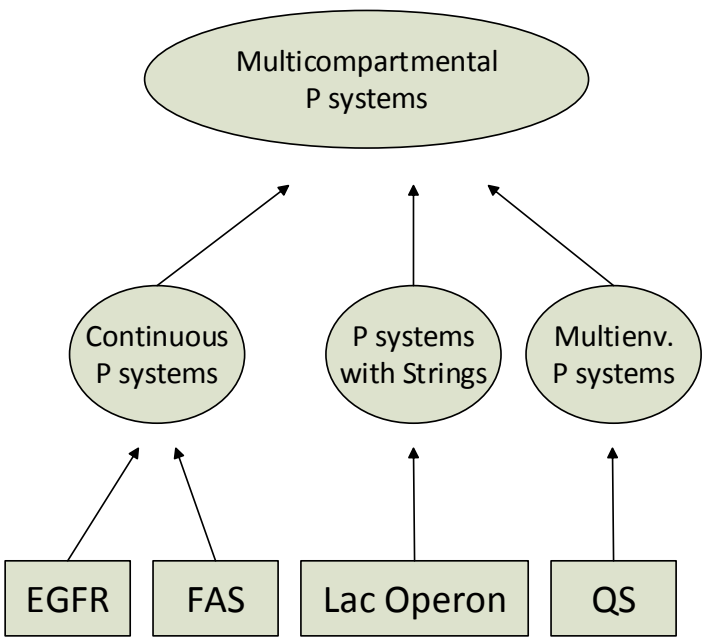
Metodología para diseñar un marco de modelización



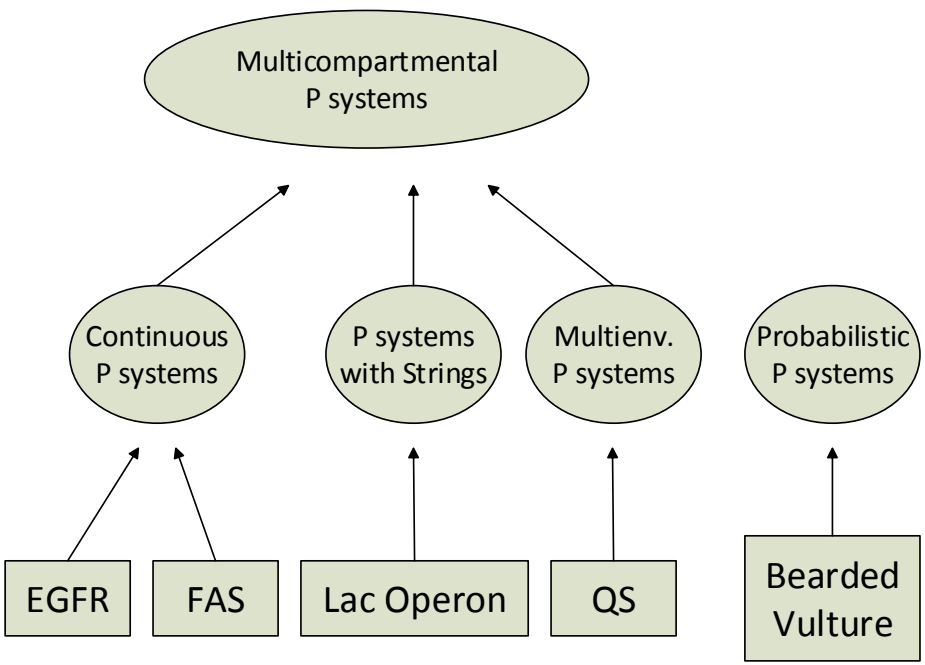
Metodología para diseñar un marco de modelización en Membrane Computing



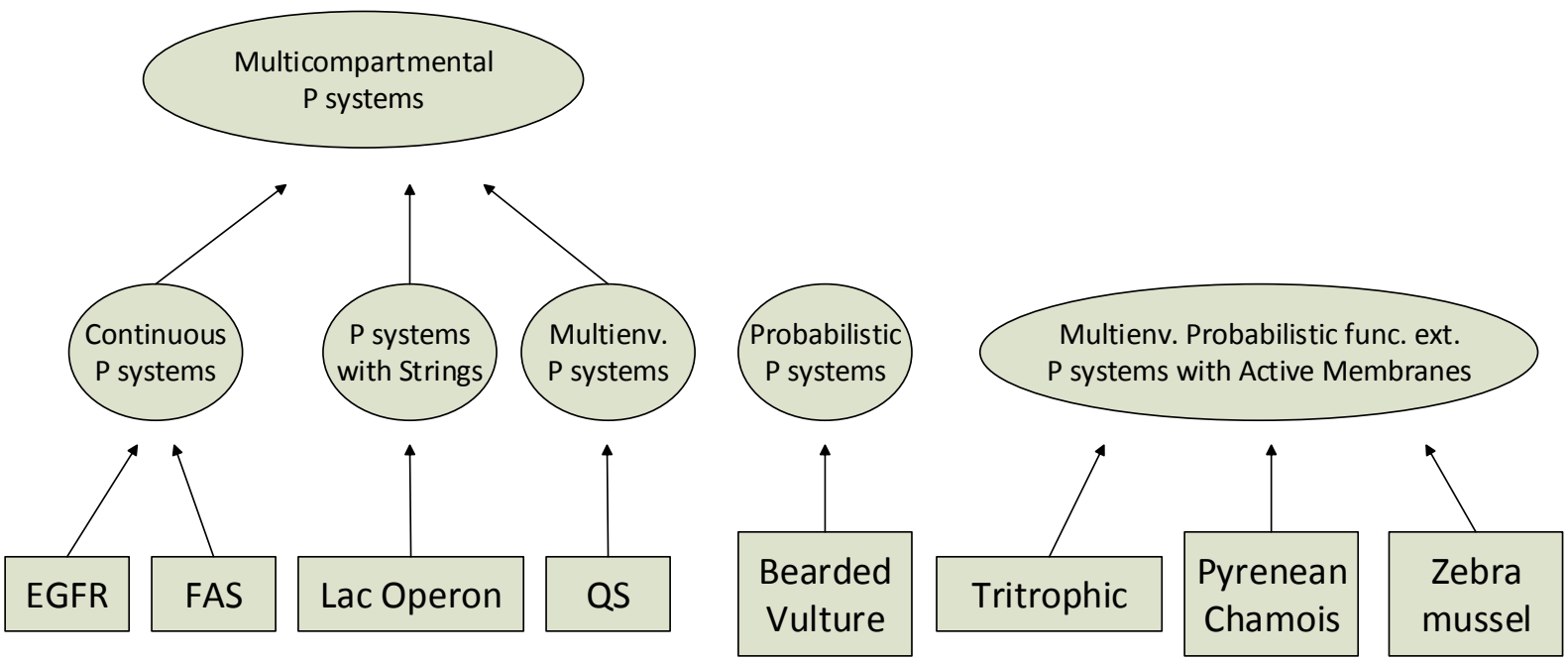
Metodología para diseñar un marco de modelización en Membrane Computing



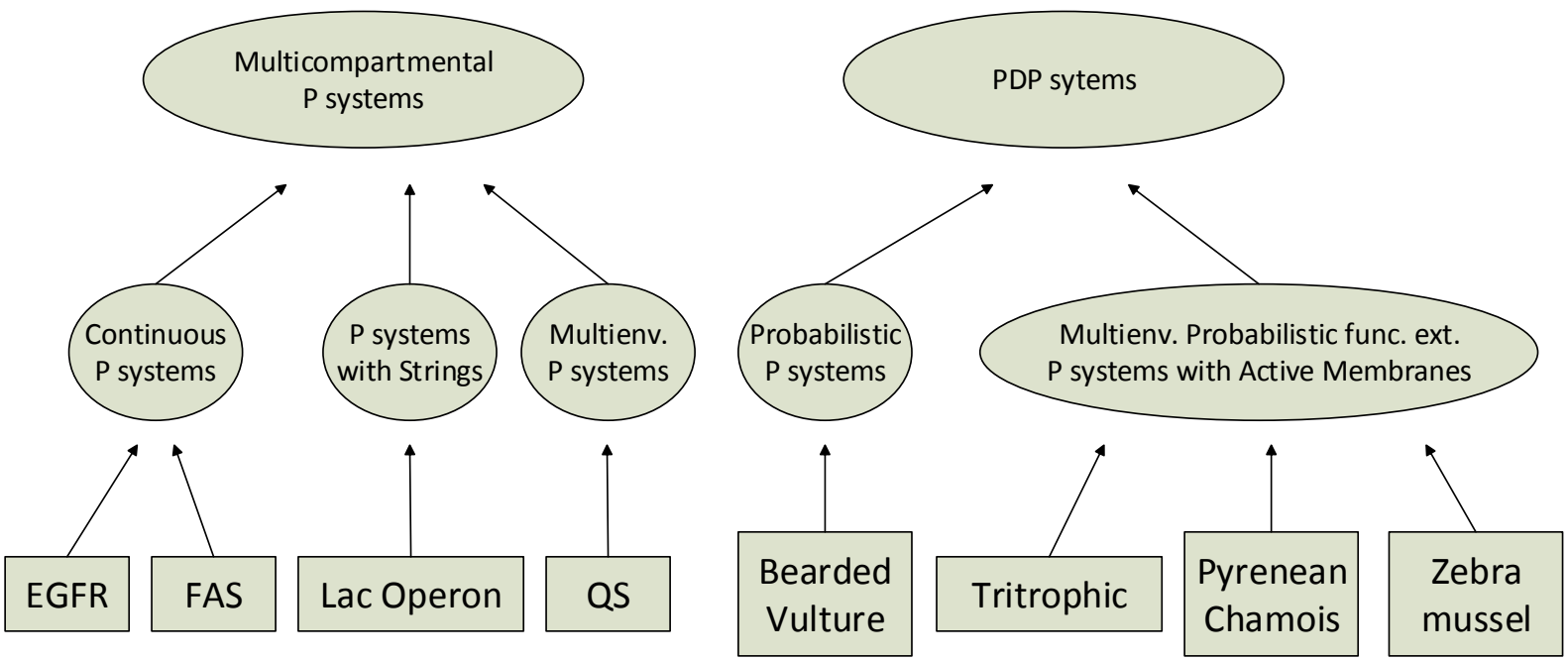
Metodología para diseñar un marco de modelización en Membrane Computing



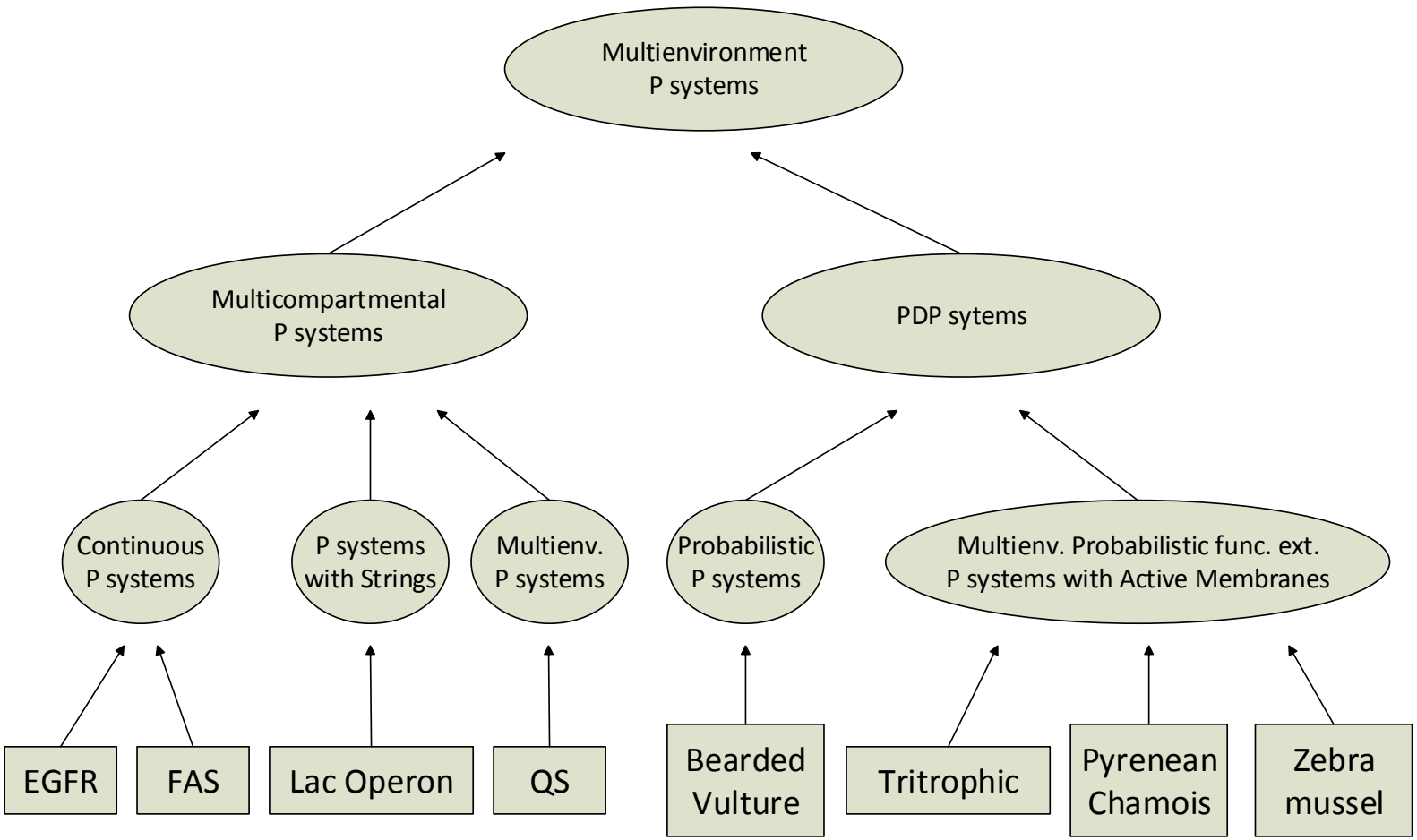
Metodología para diseñar un marco de modelización en Membrane Computing



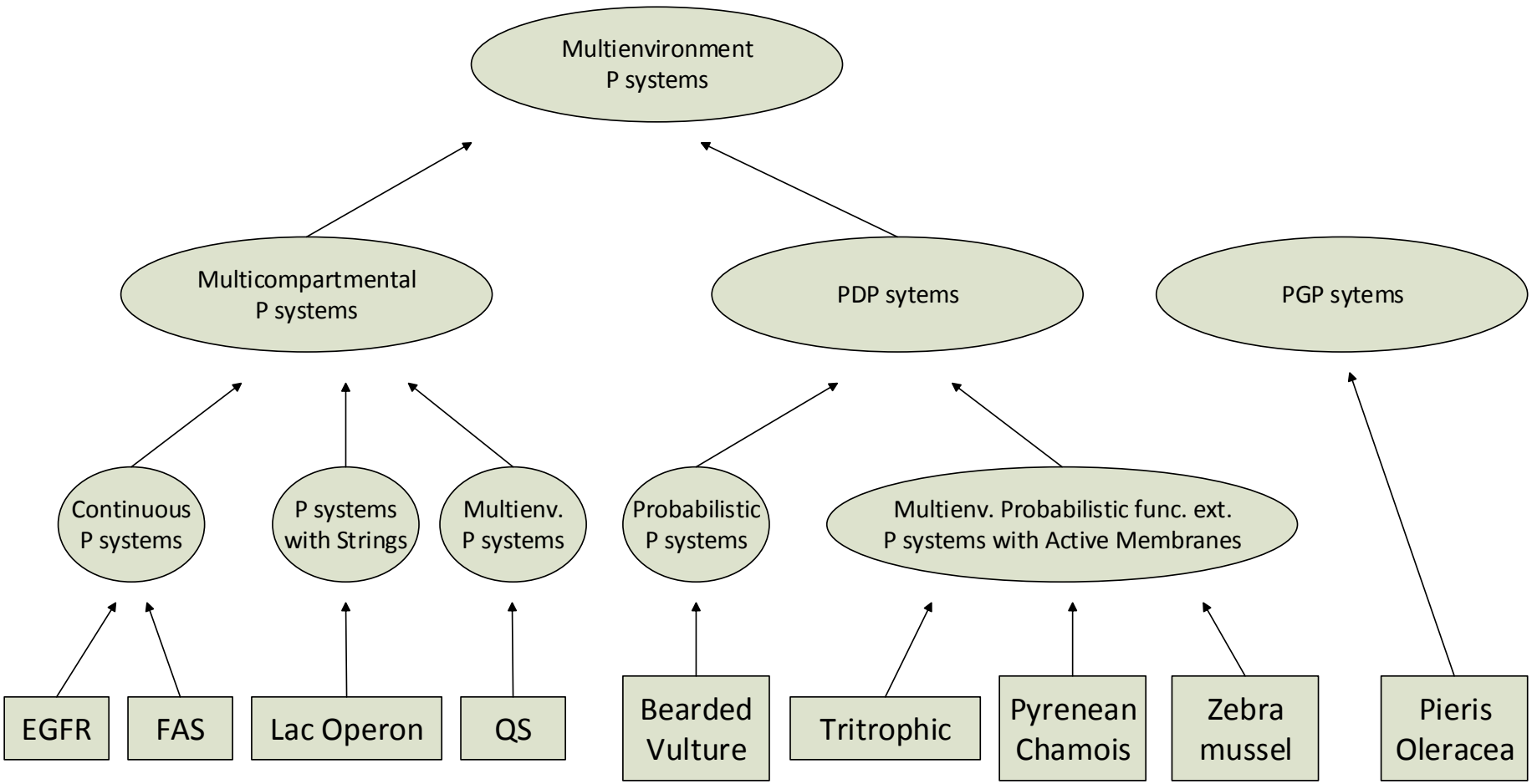
Metodología para diseñar un marco de modelización en Membrane Computing



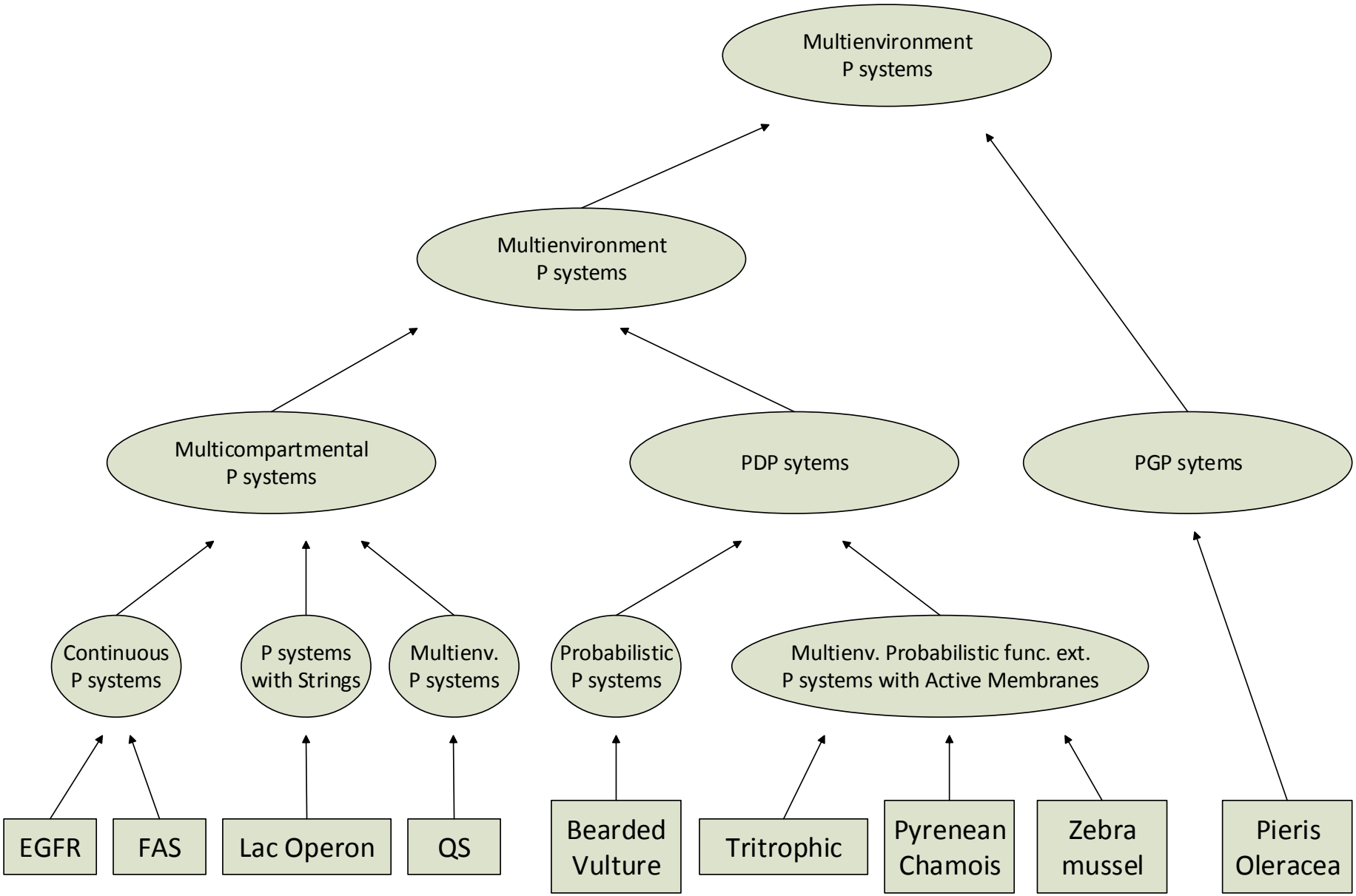
Metodología para diseñar un marco de modelización en Membrane Computing



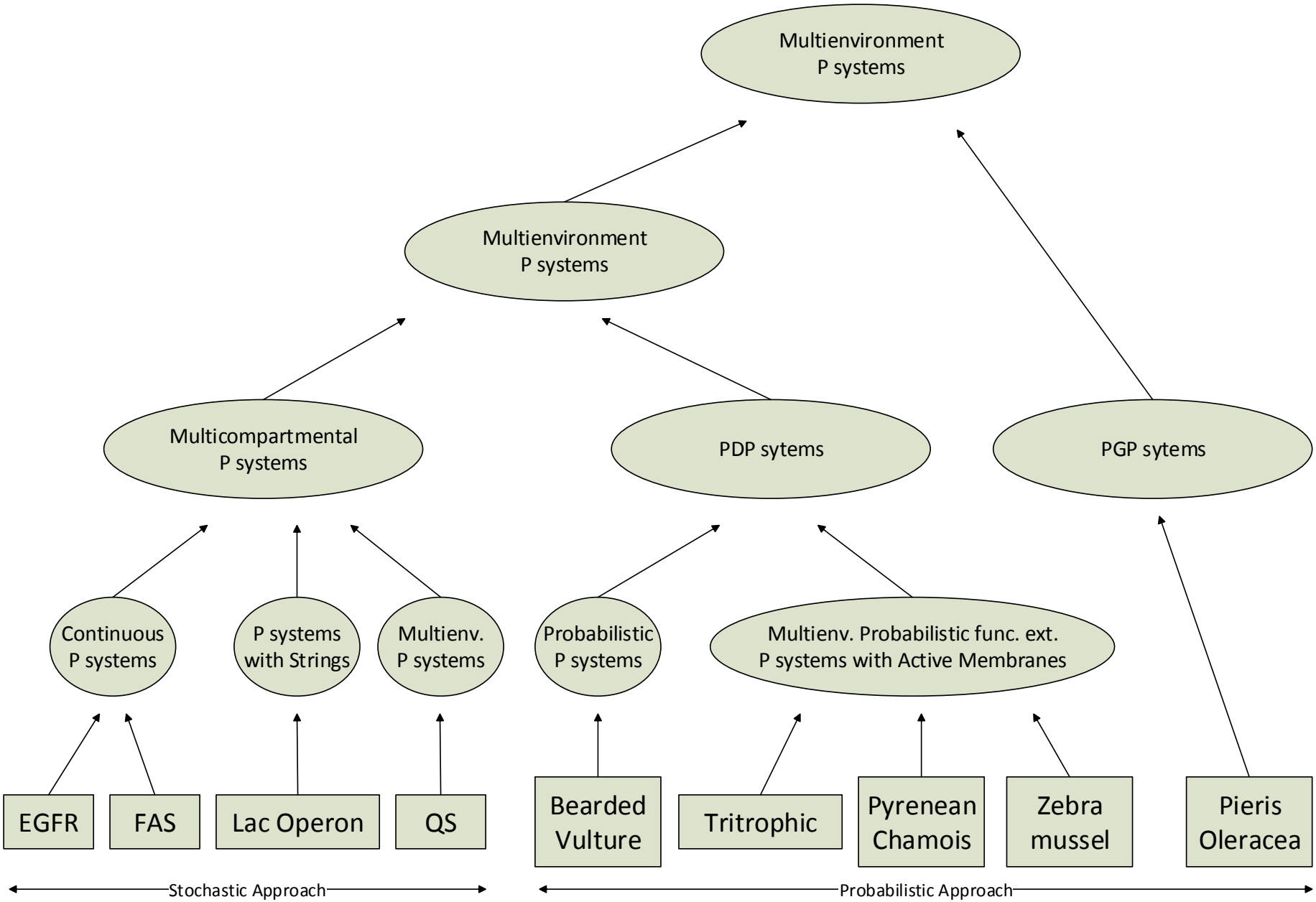
Metodología para diseñar un marco de modelización en Membrane Computing



Metodología para diseñar un marco de modelización en Membrane Computing



Metodología para diseñar un marco de modelización en Membrane Computing



Marco formal de especificación basado en sistemas de membranas

Esqueleto de un sistema de membranas de grado $q \geq 1$: $\Pi = (\Gamma, \mu, \mathcal{R})$:

- $\star$ **Γ**: alfabeto finito (de trabajo).
- $\star$ μ: es un **árbol enraizado** com q nodos (**membranas**) etiquetados biyectivamente por $1, \ldots, q$ (**1** será la **etiqueta de la piel**), y con cargas eléctricas del conjunto $\{0, +, -\}$.
- $\star$ $\mathcal{R}$: conjunto finito de reglas sobre Γ del tipo $r: \; u[\; v \;]_i^{\alpha} \rightarrow u'[\; v' \;]_i^{\alpha'}$, en donde $u, v, u', v' \in M(\Gamma)$, $i \in \{1, \ldots, q\}$, y $\alpha, \alpha' \in \{0, +, -\}$.

Es un conjunto de membranas polarizadas, jerarquizado por una estructura μ de árbol enraizado. Inicialmente, todas las membranas están con carga neutra.

Sistema P multientorno: Sintaxis

Sistema P multientorno de orden $(m, p_1, \ldots, p_m, q)$ y con T unidades de tiempo $(m, q, T \geq 1, p_j \geq 0)$:

$$\Pi = (G, \Gamma, \Sigma, \Phi, T, \{\Pi_{k,j} \mid 1 \leq k \leq p_j,\ 1 \leq j \leq m\}, \{(f_j, E_j) \mid 1 \leq j \leq m\}, \mathcal{R}_E)$$

- $G = (V, S)$ es un grafo dirigido con $m \geq 1$ nodos. Sea $V = \{e_1, \ldots, e_m\}$ (**entornos** del sistema).
- Γ, Σ y Φ son alfabetos finitos tales que $\Sigma \subseteq \Gamma$ y $\Gamma \cap \Phi = \emptyset$.
- $\Pi_{k,j} = (\Gamma, \mu, \mathcal{M}_1^{k,j}, \ldots, \mathcal{M}_q^{k,j}, \mathcal{R}, i_{in})$ son sistemas de membranas con el mismo esqueleto.
  - $\star$ μ: árbol enraizado con $q \geq 1$ nodos etiquetados biyectivamente por $\{1, \ldots, q\}$ y con cargas eléctricas de $\{0, +, -\}$.
  - $\star$ $\mathcal{M}_i^{k,j} \in M(\Gamma)$ para cada i, $1 \leq i \leq q$.
  - $\star$ $\mathcal{R}$: conjunto finito de reglas $u[v]_i^{\alpha} \xrightarrow{g} u'[v']_i^{\alpha'}$, siendo $u, v, u', v' \in M(\Gamma)$, $1 \leq i \leq q$, $\alpha, \alpha' \in \{0, +, -\}$ y g una función computable cuyo dominio es $\{0, \ldots, T\}$ que depende de e_j.
  - $\star$ i_{in}: nodo de μ.
- $f_j \in \Phi$ y $E_j \in M(\Sigma)$, para cada j, $1 \leq j \leq m$.
- $\mathcal{R}_E$ es un conjunto finito de reglas de **movimientos entre entornos**, y son del tipo:

$$(x)_{e_j} \xrightarrow{g_1} (y_1)_{e_{j_1}} \cdots (y_h)_{e_{j_h}} ; (\Pi_{k,j})_{e_j} \xrightarrow{g_2} (\Pi_{k,j})_{e_{j_1}} ; \{f\}(u)_{e_j} \xrightarrow{g_3} (v)_{e_{j_1}} ; \{f\}(u, f)_{e_j} \xrightarrow{g_4} (v, f')_{e_j}$$

donde $x, y_1, \ldots y_h \in \Sigma$, $(e_j, e_{j_i}) \in S$, $1 \leq j \leq m, 1 \leq i \leq h$, $1 \leq k \leq \max\{p_1, \ldots, p_m\}$,
$f, g \in \Phi$, $u, v \in M(\Gamma)$ y g_1, g_2, g_3, g_4 son funciones computables cuyo dominio es $\{0, \ldots, T\}$.

Un **sistema P multientorno** de orden $(m, p_1, \ldots, p_m, q)$ y con T unidades de tiempo:

- Consta de
  - m **entornos** e_j conectados entre sí por los arcos de un **grafo dirigido** G.
  - $p = p_1 + \cdots + p_m$ **sistemas P** de grado q, todos ellos con el **mismo esqueleto**.

$$\begin{array}{ll} \Pi_{1,1} \ldots \Pi_{p_1,1} & \text{(en el entorno } e_1\text{)} \\ \Pi_{1,2} \ldots \Pi_{p_2,2} & \text{(en el entorno } e_2\text{)} \\ \ldots\ldots\ldots\ldots & \\ \ldots\ldots\ldots\ldots & \\ \Pi_{1,m} \ldots \Pi_{p_m,m} & \text{(en el entorno } e_m\text{)} \end{array}$$

  - Γ alfabeto de trabajo, Σ alfabeto de objetos de los entornos y Φ alfabeto de las "banderas".
  - Un conjunto de reglas en los sistemas $\Pi_{k,j}$ que permiten su evolución.
  - Un conjunto de reglas que permiten el movimiento de objetos entre los distintos entornos.
- Inicialmente, cada entorno e_j:
  - Tiene una única bandera f_j (si el conjunto Φ es no vacío).
  - Contiene un multiconjunto E_j sobre el alfabeto Σ de objetos de los entornos.
  - Contiene los sistemas de membranas $\Pi_{1,j}, \ldots, \Pi_{p_j,j}$, todos ellos con el mismo esqueleto y sus reglas tienen asociadas funciones computables sobre $\{0, \ldots, T\}$ que dependen de e_j.
- En cada instante, un entorno e_j:
  - Tiene un una única bandera (si el conjunto Φ es no vacío).
  - Contiene objetos del alfabeto Σ.
  - Contiene algunos sistemas de membranas (los sistemas se pueden "mover" de un entorno a otro).

Sistema P multientorno: Semántica

Sistema P multientorno de orden $(m, p_1, \ldots, p_m, q)$ y con T unidades de tiempo:

Una **configuración** del sistema en un instante t está descrita por las configuraciones de los sistemas $\Pi_{k,j}$ y los pares ordenados (f_j, E_j) en ese instante.

La **aplicabilidad** de una reglas del sistema a una configuración se define de manera natural.

- **Aplicación** de una regla $u[v]_i^{\alpha} \xrightarrow{g} u'[v']_i^{\alpha'}$ a una configuración: los multiconjuntos u, v producen u', v', resp., y la carga de la membrana i pasa a ser α'.
- **Aplicación** de una regla $(x)_{e_j} \xrightarrow{g_1} (y_1)_{e_{j_1}} \cdots (y_h)_{e_{j_h}}$ a una configuración: el objeto x pasa del entorno e_j a los entornos $e_{j_1}, \ldots, e_{j_h}$ transformándose en $y_1, \ldots y_h$, respectivamente.
- **Aplicación** de una regla $(\Pi_{k,j})_{e_j} \xrightarrow{g_2} (\Pi_{k,j})_{e_{j_1}}$ a una configuración: el sistema $\Pi_{k,j}$ pasa del entorno e_j al entorno e_{j_1}.
- **Aplicación** de una regla $\{f\}\,(u)_{e_j} \xrightarrow{g_3} (v)_{e_{j_1}}$ a una configuración: el multiconjunto u produce el multiconjunto v conservándose la bandera f en e_j. Como la bandera f no se consume, esta regla **se puede aplicar múltiples veces** en un sólo paso.
- **Aplicación** de una regla $\{f\}\,(u, f)_{e_j} \xrightarrow{g_4} (v, f')_{e_j}$ a una configuración: el multiconjunto u produce el multiconjunto v y la bandera f produce la bandera f'. Como f se consume y cada entorno tiene una única bandera, esta regla **sólo se puede aplicar una vez, a lo sumo, en cada paso**.

Nota: Las funciones $g(t), g_1(t), g_2(t), g_3(t), g_4(t)$ proporcionan las "afinidades" para que unas reglas aplicables a una configuración en el instante t se puedan aplicar.

Paso de transición: se aplicará un **multiconjunto maximal de reglas** que, en su caso, sea **consistente** con respecto a las cargas de las correspondientes partes derecha.

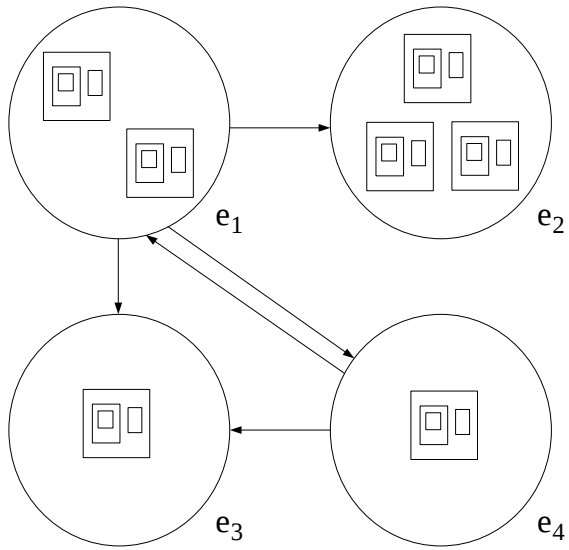
Sistema P multientorno de orden $(m, p_1, \ldots, p_m, q)$

Orientación ESTOCÁSTICA

- El conjunto Φ de las banderas es vacío.
- Los sistemas $\Pi_{k,j}$ son independientes del índice j (escribiremos simplemente Π_k, con $1 \leq k \leq p_1 + \cdots + p_m$).
- En el **instante inicial**, los sistemas Π_k son **distribuidos aleatoriamente** entre los m entornos del sistema.
- Funciones asociadas a las reglas de Π_k: **propensidades**. **No** dependen del entorno, pero **sí** del tiempo.
- Las funciones g_3, g_4 asociadas a las reglas $\{f\}(u)_{e_j} \xrightarrow{g_3} (v)_{e_{j_1}}$ y $\{f\}(u, f)_{e_j} \xrightarrow{g_4} (v, f')_{e_j}$ son constantes e iguales a 0.

Sistema P multientorno

Orientación ESTOCÁSTICA



- **Motor de inferencia**: **algoritmo multicompartimental de Gillespie** [1] y **algoritmo determinista de tiempo de espera**[1].

[1] S. Cheruku, A. Păun, F.J. Romero, M.J. Pérez-Jiménez, O.H. Ibarra. Simulating FAS-induced apoptosis by using P systems. *Progress in Natural Science*, **17**, 4 (2007), 424–431.

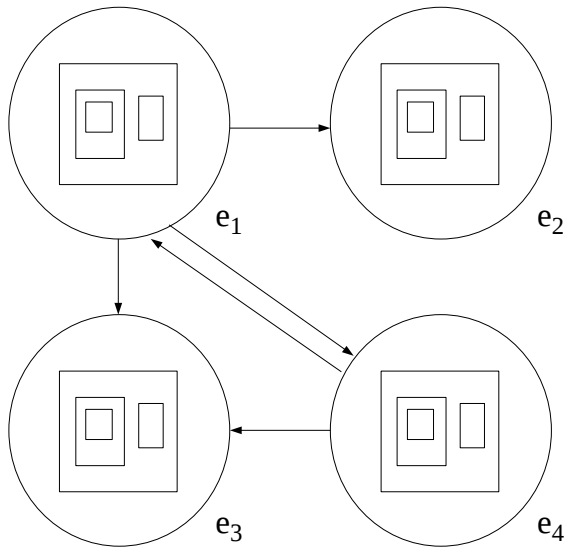
Sistema P multientorno de orden $(m, p_1, \ldots, p_m, q)$

Orientación PROBABILÍSTICA

- El número de sistemas $\Pi_{k,j}$ es, a lo sumo, el número de entornos; es decir, $p_1 + \cdots + p_m \leq m$.
- Las funciones g asociadas a las reglas $r \equiv u[v]_i^{\alpha} \xrightarrow{g} u'[v']_i^{\alpha'}$ de $\Pi_{k,j}$ son **funciones de probabilidad**: en cada instante, los valores de las funciones asociadas a las reglas de $\mathcal{R}_j$ con la misma LHS, suman 1.
- Las funciones g_1 asociadas a las reglas $(x)_{e_j} \xrightarrow{g_1} (y_1)_{e_{j_1}} \cdots (y_h)_{e_{j_h}}$ son **funciones de probabilidad**: en cada instante, los valores de las funciones asociadas a las reglas con la misma LHS, suman 1.
- Las funciones g_2 asociadas a las reglas de entornos $(\Pi_{k,j})_{e_j} \xrightarrow{g_2} (\Pi_{k,j})_{e_{j_1}}$ son constantes e iguales a 0.
- Las funciones g_3, g_4 asociadas a las reglas de entornos $\{f\}(u)_{e_j} \xrightarrow{g_3} (v)_{e_{j_1}}$ y $\{f\}(u, f)_{e_j} \xrightarrow{g_4} (v, f')_{e_j}$ son **funciones de probabilidad**: en cada instante, los valores de las funciones asociadas a las reglas con la misma LHS, suman 1.
- **No hay competencia** entre las reglas de la membrana piel de los sistemas de membranas y las de los entornos: **No existen reglas** $u[v]_1^{\alpha} \xrightarrow{g} u'[v']_1^{\alpha'}$ y reglas $(x)_{e_j} \xrightarrow{g_1} (y_1)_{e_{j_1}} \cdots (y_h)_{e_{j_h}}$ tales que $x \in u$.
- En el **instante inicial**, cada entorno e_j contiene, **a lo sumo**, un sistema $\Pi_{k,j}$.

Sistema P multientorno

Orientación PROBABILÍSTICA



- **Motor de inferencia**: **algoritmo BBB**[2], **algoritmo DNDP**[3] y **algoritmo DCBA**[4].

---

[2] M. Cardona, M.A. Colomer, A. Margalida, A. Palau, I. Pérez-Hurtado, M.J. Pérez-Jiménez, D. Sanuy. A computational modeling for real ecosystems based on P systems. *Natural Computing*, 10, 1 (2011), 39-53.

[3] M.A. Martínez, I. Pérez-Hurtado, M.J. Pérez-Jiménez, A. Riscos, M.A. Colomer. A new simulation algorithm for multienvironment probabilistic P systems. In K. Li, Z. Tang, R. Li, A.K. Nagar, R. Thamburaj (eds.) *Proceedings 2010 IEEE Fifth International Conference on Bio-inpired Computing: Theories and Applications*, IEEE Press, Volume 1, September 23-26, 2010, Changsha, China, pp. 59-68.

[4] M.A. Martínez, I. Pérez, M. García–Quismondo, L.F. Macías, L. Valencia, A. Romero, C. Graciani, A. Riscos, M.A. Colomer, M.J. Pérez-Jiménez. DCBA: Simulating population dynamics P systems with proportional objects distribution. *Lecture Notes in Computer Science*, 7762 (2013), 257-276

Sistema P multicompartimental

Es un sistema P multientorno de orden $(m, p_1, \ldots, p_m, , q)$ con **orientación estocástica**:

$$\Pi = (G, \Gamma, \Sigma, \Phi, T, \{\Pi_k \mid 1 \leq k \leq p_1 + \cdots + p_m, \}, \{E_j \mid 1 \leq j \leq m\}, \mathcal{R}_E)$$

- $G = (V, S)$ es un grafo dirigido con $m \geq 1$ nodos. Sea $V = \{e_1, \ldots, e_m\}$.
- Γ y Σ son alfabetos finitos tales que $\Sigma \subsetneq \Gamma$. Además, $\boxed{\Phi = \emptyset}$.
- $T \geq 1$ y $p_1 + \cdots + p_m \geq 1$ son números naturales.
- Para cada $k, 1 \leq k \leq p_1 + \cdots + p_m$, $\Pi_k = (\Gamma, \mu, \mathcal{M}_1^k, \ldots, \mathcal{M}_q^k, \mathcal{R}, i_{in})$ es un sistema de membranas tal que todos ellos tienen el mismo esqueleto:
  - μ: árbol enraizado con $q \geq 1$ nodos etiquetados biyectivamente por $\{1, \ldots, q\}$ y con cargas eléctricas de $\{0, +, -\}$.
  - $\mathcal{M}_i^k \in M(\Gamma)$ para cada i, $1 \leq i \leq q$.
  - $\mathcal{R}$: conjunto finito de reglas $u[v]_i^{\alpha} \xrightarrow{g} u'[v']_i^{\alpha'}$, siendo $u, v, u', v' \in M(\Gamma)$, $1 \leq i \leq q$, $\alpha, \alpha' \in \{0, +, -\}$ y g una función computable cuyo dominio es $\{0, \ldots, T\}$ que representa la **propensidad**.
  - i_{in}: nodo de μ.
- $E_j \in M(\Sigma)$ para cada j, $1 \leq j \leq m$.
- $\mathcal{R}_E$ es un conjunto finito de reglas de movimientos entre entornos, y son del tipo:

$$(x)_{e_j} \xrightarrow{g_1} (y_1)_{e_{j_1}} \cdots (y_h)_{e_{j_h}} \; ; \; (\Pi_k)_{e_j} \xrightarrow{g_2} (\Pi_k)_{e_{j_1}}$$

donde $x, y_1, \ldots y_h \in \Sigma$, $(e_j, e_{j_i}) \in S$, $1 \leq j \leq m, 1 \leq i \leq h$, $1 \leq k \leq p_1 + \cdots + p_m$ y g_1, g_2 son **propensidades** cuyo dominio es $\{0, \ldots, T\}$ (no dependen del entorno e_j).

Un **sistema P multicompartimental** de orden $(m, p_1, \ldots, p_m, q)$:

- Consta de
  - m **entornos** conectados entre sí por los arcos de un grafo dirigido G.
  - $p_1 + \cdots + p_m$ **sistemas de membranas**, Π_k, de orden q (todos ellos con el mismo **esqueleto**).
- En el **instante inicial**, los sistemas Π_k son **distribuidos aleatoriamente** entre los m entornos del sistema.
- En cada instante, un entorno e_j sólo puede contener **objetos de Σ** y algunos **sistemas de membranas**.
- **Aplicabilidad** de una regla $u[v]_i^{\alpha} \xrightarrow{g} u'[v']_i^{\alpha'}$: mediante su aplicación, los multiconjuntos u, v producen u', v', resp., y la carga de i pasa a ser α'.
- **Aplicabilidad** de una regla $(x)_{e_j} \xrightarrow{g_1} (y_1)_{e_{j_1}} \cdots (y_h)_{e_{j_h}}$: mediante su aplicación, el objeto x pasa del entorno e_j a los entornos $e_{j_1}, \ldots, e_{j_h}$ transformándose en $y_1, \ldots y_h$, respectivamente
- **Aplicabilidad** de una regla $(\Pi_k)_{e_j} \xrightarrow{g_2} (\Pi_k)_{e_{j'}}$: mediante su aplicación, el sistema de membranas Π_k pasa de e_j a $e_{j'}$

Nota 1: $g(t), g_1(t), g_2(t)$ proporcionan las "afinidades" para que reglas aplicables en un instante t se puedan aplicar en ese instante (**propensidades**: **sólo** dependen de las multiplicidades de los objetos en ese instante)

Nota 2: **Motor de inferencia**: **algoritmo multicompartimental de Gillespie** o **algoritmo determinista de tiempo de espera**.

Algoritmo multicompartimental de Gillespie

Los sistemas P multicompartimentales constan de:

- ⋆ Un conjunto de entornos interconectados por un grafo dirigido.
  - ∗ Cada entorno delimita una región (volumen) que contiene su propio conjunto de reglas y multiconjunto de objetos.
- ⋆ Unos sistemas P que trabajan a modo de células y que están contenidos en los entornos.
  - ∗ A través de las membranas, cada sistema P delimita regiones (volúmenes) y cada una de ellas tiene su propio conjunto de reglas y multiconjunto de objetos.

Los entornos y las membranas de los sistemas P contenidos en los entornos se denominarán, genéricamente, **compartimentos**.

El algoritmo de Gillespie proporciona una adecuada simulación estocástica de procesos biomoleculares que tienen lugar en **un medio/volumen fijo**.

Para capturar la semántica de los sistemas P multicompartimentales:

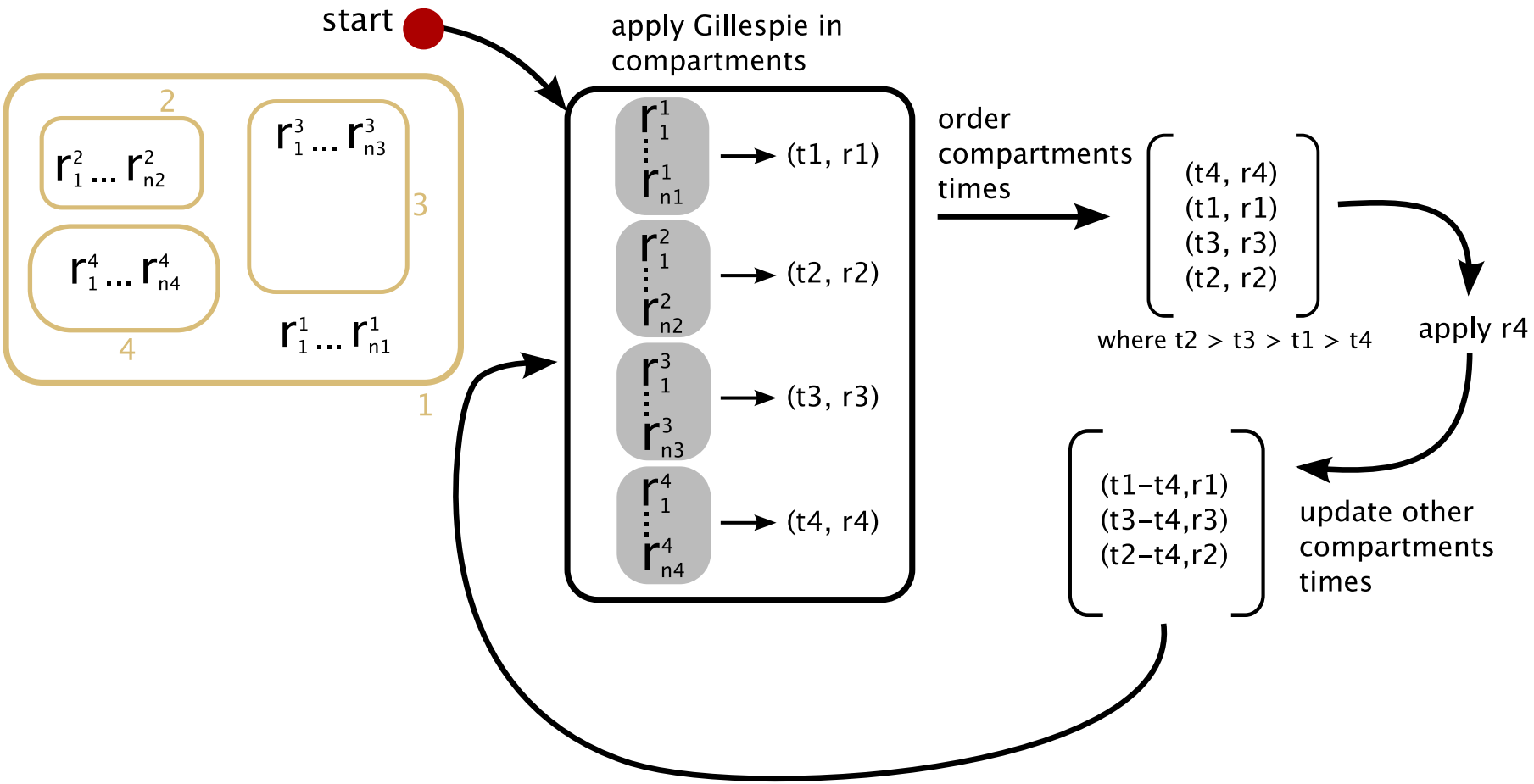
- ∗ Vamos a generalizar dicho algoritmo permitiendo que cada compartimento evolucione de acuerdo con el algoritmo "ordinario" de Gillespie.

Algoritmo multicompartimental de Gillespie

Entrada: Un sistema P multicompartimental y las propensidades iniciales de las reglas.

- **Inicialización:**
  - ⋆ Inicializar el tiempo de simulación: $t = 0$.
  - ⋆ Para cada compartimento (membranas y entornos) c hacer:
    - Ejecutar el algoritmo de Gillespie clásico en c, devolviendo (j_c, τ_c).
  - ⋆ Ordenar la lista de ternas (j_c, τ_c, c) con relación a τ_c (en orden decreciente).
- **Iteración:**
  - ⋆ Elegir la primera terna de la lista: $(j_{c_0}, \tau_{c_0}, c_0)$.
  - ⋆ Actualizar el tiempo de simulación: $t \longleftarrow t + \tau_{c_0}$.
  - ⋆ Actualizar el tiempo de espera para el resto de las ternas, restando τ_{c_0}.
  - ⋆ Aplicar la regla $r_{j_{c_0}}$ una sóla vez actualizando los compartimentos afectados.
  - ⋆ Para cada compartimento c' afectado por la aplicación de la regla $r_{j_{c_0}}$ hacer
    - eliminar la terna $(j_{c'}, \tau_{c'}, c')$ de la lista;
    - actualizar las propensidades;
    - ejecutar el algoritmo ordinario de Gillespie en c', obteniendo $(j^\star_{c'}, \tau^\star_{c'}, c')$;
    - añadir la terna $(j^\star_{c'}, \tau^\star_{c'}, c')$ a la lista;
    - ordenar la nueva lista (en orden decreciente según el tiempo de espera τ).
  - ⋆ Iterar el proceso mientras no se verifique la condición de parada.
- **Finalización:**
  - ⋆ Si el tiempo de simulación t excede al tiempo máximo prefijado, T, finalizar.

Algoritmo multicompartimental de Gillespie



Algoritmo determinista con tiempo de espera

El algoritmo multicompartimental de Gillespie:

- Captura la semántica de modelos estocásticos de procesos biomoleculares basados en sistemas P.
- Se usa para estudiar sistemas celulares en donde las aproximaciones deterministas y/o continuas **no** son adecuadas.
- Es **muy costoso** desde el punto de vista computacional.

En el caso de procesos en donde el número de invididuos es *elevado*, las aproximaciones deterministas y/o continuas pueden ser "aceptables".

⋆ Se propone un **algoritmo determinista con tiempo de espera**.

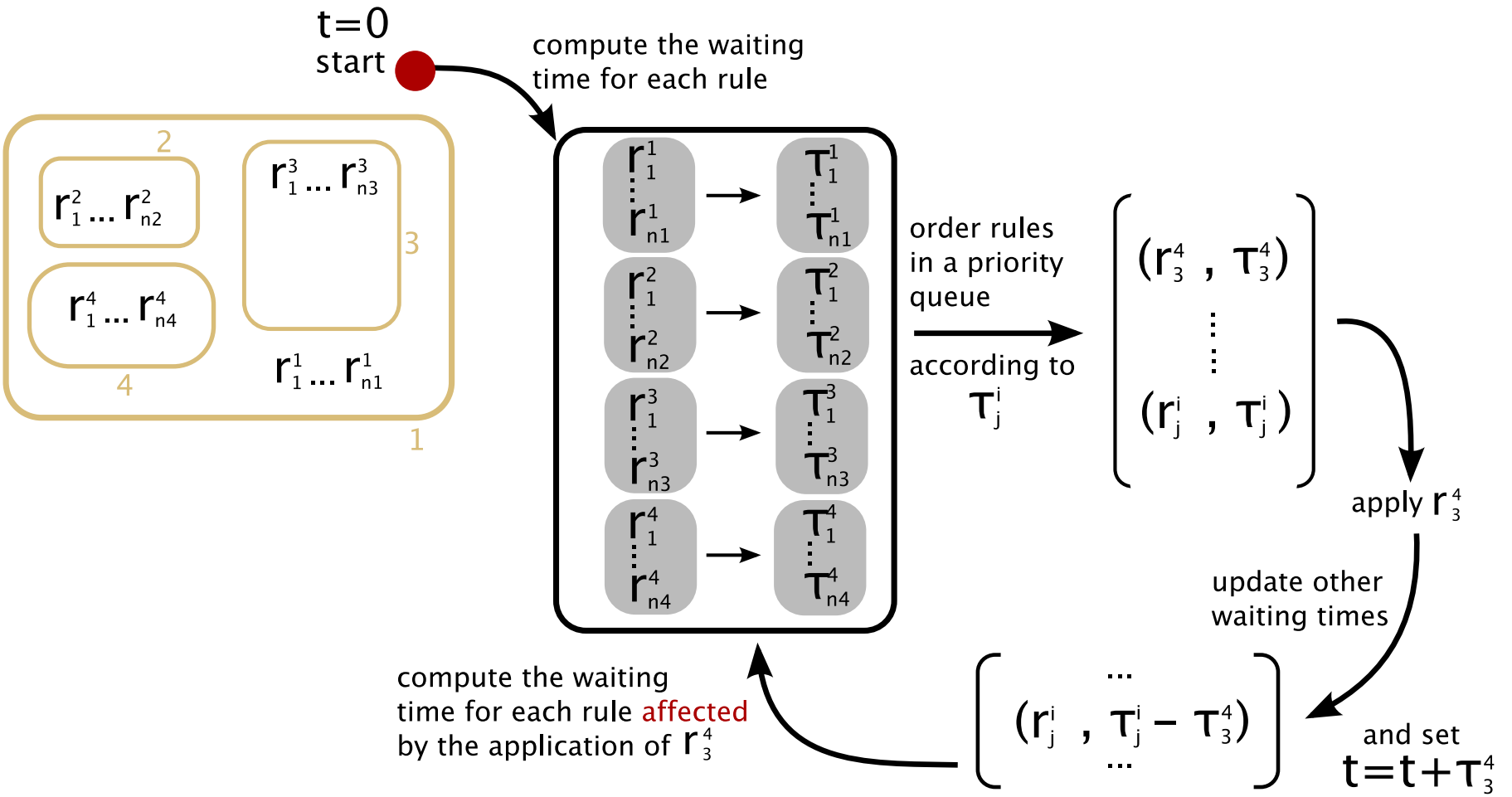
- Es una versión determinista del algoritmo multicompartimental de Gillespie.
- Cada regla del sistema (reacción química) r tiene asociada:
  - ⋆ una **velocidad** v_r que es el producto de la constante estocástica de r por las concentraciones de los reactantes (ley de acción de masas);
  - ⋆ un **tiempo de espera** $\tau_r = \frac{1}{v_r}$.

Algoritmo determinista con tiempo de espera

Entrada: Un sistema P multicompartimental y las propensidades iniciales de las reglas.

- **Inicialización:**
  - ⋆ Inicializar el tiempo de simulación: $t = 0$.
  - ⋆ Para cada regla r_i de cada compartimento c, calcular la terna (i, τ_i, c), siendo τ_i el tiempo de espera asociado a la regla r_i.
  - ⋆ Ordenar la lista de ternas (i, τ_i, c) de acuerdo a los tiempos de espera (en orden decreciente).
- **Iteración:**
  - ⋆ Elegir la primera terna de la lista: (i_0, τ_{i_0}, c_0).
  - ⋆ Actualizar el tiempo de simulación: $t \longleftarrow t + \tau_{i_0}$.
  - ⋆ Actualizar el tiempo de espera para el resto de las ternas, restando τ_{i_0}.
  - ⋆ Aplicar la regla r_{i_0} una sóla vez actualizando los compartimentos afectados.
  - ⋆ Eliminar la terna (i_0, τ_{i_0}, c_0) de la lista.
  - ⋆ Para cada compartimento c' afectado por la aplicación de la regla r_{i_0} hacer
    - eliminar las ternas correspondientes a las reglas de c';
    - actualizar las multiplicidades de los objetos presentes en c';
    - calcular los tiempos de espera de las reglas de c';
    - añadir las nuevas ternas correspondientes a las reglas de c';
    - ordenar la nueva lista (en orden decreciente según el tiempo de espera τ).
  - ⋆ Iterar el proceso mientras no se verifique la condición de parada.
- **Finalización:**
  - ⋆ Si el tiempo de simulación t excede al tiempo máximo prefijado, T, finalizar.

Algoritmo determinista con tiempo de espera



Comparación de ambos algoritmos

* En el **algoritmo multicompartimental de Gillespie**:
  - Se selecciona un par (**regla**, **tiempo de espera**) de cada compartimento: **algoritmo ordinario de Gillespie**.
  - Existen tantas ternas (**regla**, **tiempo de espera**, **compartimento**) como **compartimentos**.
* En el **algoritmo determinista con tiempo de espera**:
  - Se selecciona una terna (**regla**, **tiempo de espera**, **compartimento**) por cada regla del sistema: **ley de acción de masas**.
  - Existen tantas ternas (**regla**, **tiempo de espera**, **compartimento**) como **reglas**.
* En general, el algoritmo determinista con tiempo de espera es **más eficiente** que el algoritmo multicompartimental de Gillespie.

Population Dynamics P systems (PDP)

Son sistemas P multientorno de orden $(m, p_1, \ldots, p_m, q)$ con **orientación probabilística**:

$$\Pi = (G, \Gamma, \Sigma, \Phi, T, \{\Pi_k \mid 1 \leq k \leq p_1 + \cdots + p_m\}, \{E_j \mid 1 \leq j \leq m\}, \mathcal{R}_E)$$

- $\mathbf{p_j = 1}$, $1 \leq j \leq m$,y $\mathbf{m = p_1 + \cdots + p_m}$ (**inicialmente**, cada entorno e_j contiene **exactamente un** sistema Π_k).
- $G = (V, S)$ es un grafo dirigido con $m \geq 1$ nodos. Sea $V = \{e_1, \ldots, e_m\}$.
- Γ y Σ son alfabetos finitos tales que $\Sigma \subsetneq \Gamma$. Además, $\boxed{\Phi = \emptyset}$.
- $T \geq 1$ es un número natural.
- $\Pi_k = (\Gamma, \mu, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}, i_{in})$ son sistemas de membranas con el mismo esqueleto.
  - μ: árbol enraizado con $q \geq 1$ nodos etiquetados biyectivamente por $\{1, \ldots, q\}$ y con cargas eléctricas de $\{0, +, -\}$.
  - $\mathcal{M}_i \in M(\Gamma)$ para cada i, $1 \leq i \leq q$.
  - $\mathcal{R}$: conjunto finito de reglas $u[v]_i^{\alpha} \xrightarrow{g} u'[v']_i^{\alpha'}$; en donde $u, v, u', v' \in M(\Gamma)$, $1 \leq i \leq q$, $\alpha, \alpha' \in \{0, +, -\}$ y g es una **función de probabilidad** con dominio $\{0, \ldots, T\}$. En cada instante, la suma de las probabilidades asociadas a las reglas con la misma LHS, vale 1.
  - i_{in}: nodo de μ.
- $E_j \in M(\Sigma)$, para cada j, $1 \leq j \leq m$.
- $\mathcal{R}_E$ es un conjunto finito de reglas de entornos $(x)_{e_j} \xrightarrow{g_1} (y_1)_{e_{j_1}} \cdots (y_h)_{e_{j_h}}$, donde $x, y_1, \ldots y_h \in \Sigma$, $(e_j, e_{j_i}) \in S$, $1 \leq j \leq m, 1 \leq i \leq h$ y g_1 es una **función de probabilidad** cuyo dominio es $\{0, \ldots, T\}$. En cada instante, la suma de las probabilidades asociadas a las reglas con la misma LHS, vale 1.
- **No hay competencia** entre las reglas de la membrana piel de los sistemas de membranas y las de los entornos: No existen reglas $u[v]_1^{\alpha} \xrightarrow{g} u'[v']_1^{\alpha'}$ y reglas $(x)_{e_j} \xrightarrow{g_1} (y_1)_{e_{j_1}} \cdots (y_h)_{e_{j_h}}$ tales que $x \in u$.

Diremos que el **sistema PDP** es **de orden (m, q)** y notaremos $(\Gamma, \Sigma, T, E_1, \Pi_1 \ldots, E_m, \Pi_m \mathcal{R}_E, p_{\mathcal{R}_E})$, en donde $p_{\mathcal{R}_E}$ proporciona las probabilidades asociadas a las reglas de los entornos.

Sistema PDP

Sistema PDP de orden (m, q): conjunto de m entornos $e_1, \ldots, e_m$ interconectados por los arcos de un grafo dirigido G tales que cada entorno contiene un único sistema de membranas "ordinario" (todos con el mismo esqueleto).

- Cada entorno e_j sólo puede contener símbolos del alfabeto Σ y, además, un único sistema de membranas $\Pi_k = (\Gamma, \mu, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}, i_{in})$, tal que:

  (a) Los multiconjuntos iniciales de Π_k dependen del entorno e_j.

  (b) Las funciones de probabilidad asociadas a las reglas de Π_k dependen del entorno e_j.

- Las funciones de probabilidad asociadas a las reglas del entorno e_j también dependen de dicho entorno.

Semántica de los sistemas PDP: se implementa mediante algoritmos de simulacion.

Probabilistic Guarded P systems (PGP)

Son sistemas P multientorno $(m, p_1, \ldots, p_m, q)$ con **orientación probabilística**:

$$\Pi = (G, \Gamma, \Sigma, \Phi, T, \{\Pi_{k,j} \mid 1 \leq k \leq p_j,\ 1 \leq j \leq m\}, \{(f_j, E_j) \mid 1 \leq j \leq m\}, \mathcal{R}_E)$$

- $G = (V, S)$ es un grafo dirigido con $m \geq 1$ nodos. Sea $V = \{e_1, \ldots, e_m\}$ (**entornos** del sistema).
- Γ, Σ y Φ son alfabetos finitos tales que $\Sigma \cap \Phi = \emptyset$ y $\boxed{\Gamma = \Sigma}$.
- $T \geq 1$ es un número natural y $\boxed{p_j = 0}$, para cada $1 \leq j \leq m$: **no existen sistemas de membranas** $\Pi_{k,j}$.
- Para cada j, $1 \leq j \leq m$, $f_j \in \Phi$ y $E_j \in M(\Sigma)$.
- $\mathcal{R}_E$ es un conjunto finito de reglas de entornos del tipo:
  - $\{f\}\,(u)_{e_j} \xrightarrow{g_3} (v)_{e_{j_1}}$.
  - $\{f\}\,(u, f)_{e_j} \xrightarrow{g_4} (v, g)_{e_j}$.

  En donde $f, g \in \Phi$, $u, v \in M(\Sigma)$, $1 \leq j, j_1 \leq m$, $(e_j, e_{j_1}) \in S$ y g_3, g_4 son funciones de probabilidad cuyo dominio es $\{0, \ldots, T\}$. Además, para cada $f \in \Phi$, $u \in M(\Sigma)$ y $1 \leq j \leq m$ se verifica que:
  - La suma de las probabilidades asociadas a las reglas cuya LHS es $\{f\}\,(u)_{e_j}$, vale 1.
  - Sólo existe una regla cuya LHS es $\{f\}\,(u, f)_{e_j}$. Además, la probabilidad asociada a esa regla es 1.

Diremos que el **sistema PGP** es **de orden m** y notaremos $(\Sigma, \Phi, T, (f_1, E_1), \ldots, (f_m, E_m), \mathcal{R}_E, p_{\mathcal{R}_E})$, pues G viene dado implícitamente por $\mathcal{R}_E$ y $p_{\mathcal{R}_E}$ proporciona las probabilidades asociadas a las reglas de los entornos.

Sistema PGP

Sistema PGP de orden m: $\Pi = (\Sigma, \Phi, (f_1, E_1), \ldots, (f_m, E_m), \mathcal{R}_E, p_{\mathcal{R}_E})$

- Conjunto de m entornos (que denominaremos **células**) interconectados por los arcos de un grafo, dado implícitamente a través del conjunto de reglas $\mathcal{R}_E$.
- En cada instante, un entorno e_j contiene **una bandera** (elemento de Φ) que está **sobre** los entornos. Inicialmente, la bandera del entorno e_j es f_j.
- En cada instante, un entorno e_j contiene un multiconjunto de **objetos** sobre Σ que está **en el interior/dentro de** e_j. Inicialmente, el multiconjunto de objetos del entorno e_j es E_j.
- Existe un conjunto finito $\mathcal{R}_E$ de reglas que son del tipo:
  - ⋆ $\{f\}\,(u)_{e_j} \longrightarrow (v)_{e_{j_1}}$ (esta regla "determina" un arco $e_j \to e_{j_1}$).
  - ⋆ $\{f\}\,(u, f)_{e_j} \longrightarrow (v, g)_{e_j}$ (esta regla "determina" un arco $e_j \to e_j$).

  En donde $f, g \in \Phi$, $u, v \in M(\Sigma)$ y $1 \leq j, j_1 \leq m$. Estas reglas satisfacen las condiciones antes descritas.
- $p_{\mathcal{R}_E}$ es una función que asigna una probabilidad a cada regla del sistema.

Semántica de los PGP systems

Configuración de $\Pi = (\Sigma, \Phi, (f_1, E_1), \ldots, (f_m, E_m), \mathcal{R}_E, p_{\mathcal{R}_E})$ en un instante t:

- Una tupla $(x_1, u_1, \ldots, x_m, u_m)$, donde $x_i \in \Phi$ y $u_i \in M(\Sigma)$, $1 \leq i \leq m$.
- **Configuración inicial** de Π: $(f_1, E_1, \ldots, f_m, E_m)$.
- **Aplicabilidad** de una regla $\{f\}\,(u)_{e_j} \xrightarrow{g_3} (v)_{e_{j_1}}$ a una configuración $\mathcal{C}_t = (x_1, u_1, \ldots, x_m, u_m)$ del sistema en un instante t: ha de verificarse $x_j = f$ y $u \subseteq u_j$.
  - ⋆ Mediante su aplicación se elimina el multiconjunto u de e_j y produce el multiconjunto v en e_{j_1}.
- **Aplicabilidad** de una regla $\{f\}\,(u, f)_{e_j} \xrightarrow{g_4} (v, g)_{e_j}$ a una configuración $\mathcal{C}_t = (x_1, u_1, \ldots, x_m, u_m)$ del sistema en un instante t: ha de verificarse que $x_j = f$ y $u \subseteq u_j$.
  - ⋆ Mediante su aplicación se cambia la bandera f de e_j por la bandera g y se reemplaza el multiconjunto u por el multiconjunto v en e_j. En cada instante sólo existe una bandera en cada entorno, por tanto **esta regla sólo se podrá aplicar**, a lo sumo, **una vez** en un instante dado.
- A partir de aquí se definen los conceptos semánticos habituales: configuración de parada, paso de transición, computación del sistema, computación de pàrada y resultado de una computación.

Sistema P multientornos

$$\left.\begin{array}{ll} \text{Alfabeto de } \textbf{TRABAJO}: & \Gamma \\ \text{Alfabeto de los } \textbf{ENTORNOS}: & \Sigma \\ \text{Alfabeto de las } \textbf{BANDERAS}: & \Phi \end{array}\right\} \Sigma \subseteq \Gamma \quad \text{y} \quad \Gamma \cap \Phi = \emptyset$$

$\mathbf{p_1} + \ldots + \mathbf{p_m}$ sistemas P, todos ellos de orden $\mathbf{q}$.

- ⋆ Reglas de los sistemas de membranas:
  - $u[v]_i^{\alpha} \xrightarrow{g(t)} u'[v']_i^{\alpha'}$ (**Tipo 0**).

$\mathbf{m}$ entornos.

- ⋆ Reglas de los entornos:
  - **Tipo 1**: $(x)_{e_j} \xrightarrow{g_1(t)} (y_1)_{e_{j_1}} \cdots (y_h)_{e_{j_h}}$.
  - **Tipo 2**: $(\Pi_{k,j})_{e_j} \xrightarrow{g_2(t)} (\Pi_{k,j})_{e_{j'}}$.
  - **Tipo 3**: $\{f\}\,(u)_{e_j} \xrightarrow{g_3(t)} (v)_{e_{j_1}}$.
  - **Tipo 4**: $\{f\}\,(u,f)_{e_j} \xrightarrow{g_4(t)} (v,g)_{e_j}$.

Sistema P multicompartimentales - PDP - PGP

⋆ SISTEMAS P MULTICOMPARTIMENTALES

- ♣ El alfabeto de los entornos está estríctamente contenido en el alfabeto de trabajo: $\Sigma \subsetneq \Gamma$.
- ♣ **NO** hay banderas: $\Phi = \emptyset$.
- ♣ Sólo hay reglas de los tipos 0, 1 y 2 (es decir, $g_3(t) = g_4(t) = 0$).
- ♣ Inicialmente, los sistemas de membranas están distribuidos aleatoriamente entre los $\mathbf{m}$ entornos.

⋆ SISTEMAS PDP

- ♣ El alfabeto de los entornos está estríctamente contenido en el alfabeto de trabajo: $\Sigma \subsetneq \Gamma$.
- ♣ **NO** hay banderas: $\Phi = \emptyset$.
- ♣ Sólo hay reglas de los tipos 0 y 1 (es decir, $g_2(t) = g_3(t) = g_4(t) = 0$).
- ♣ Inicialmente, en cada entorno existe **un único** sistema P; es decir $\mathbf{p_1} + \cdots + \mathbf{p_m} = \mathbf{m}$.

⋆ SISTEMAS PGP

- ♣ El alfabeto de los entornos coincide con el alfabeto de trabajo: $\Sigma = \Gamma$.
- ♣ **NO** hay sistemas P; es decir, $\mathbf{p_j} = \mathbf{0}, 1 \leq j \leq m$.
- ♣ Sólo hay reglas de los tipos 3 y 4 (es decir, $g(t) = g_1(t) = g_2(t) = 0$).

PDP: algoritmos de simulación

Binomial Block Based simulation algorithm (**BBB** [5])

Estrategia basada en:

- ⋆ Distribución binomial.
- ⋆ El concepto de **bloque** de reglas:
  - ♣ Conjunto de reglas con la misma parte izquierda.
  - ♣ Dada una regla r, notaremos $l(r)$ la LHS de dicha regla y $r(r)$ la RHS.

Cada paso de computación se estructura en dos etapas:

- ⋆ Fase de selección de reglas.
- ⋆ Fase de ejecución de reglas.

[5] M. Cardona, M.A. Colomer, A. Margalida, A. Palau, I. Pérez-Hurtado, M.J. Pérez-Jiménez, D. Sanuy. A computational modeling for real ecosystems based on P systems. *Natural Computing*, 10, 1 (2011), 39-53.

Esquema algorítmico del BBB

Entrada: Un sistema PDP de orden (m, q), con T unidades de tiempo (su configuración inicial se notará por C_0)

for $t \leftarrow 0$ to T **do**

$C_t \leftarrow$ configuración del sistema en el instante t

Inicialización

Fase de selección: genera un multiconjunto de reglas aplicables.

Fase de ejecución: ejecuta las reglas seleccionadas

$C_{t+1} \leftarrow C_t$

end for

Inicialización del BBB

- $\mathcal{B}_\Pi \leftarrow$ conjunto ordenado de bloques de Π (orden aleatorio)
- En cada bloque se considera un **orden aleatorio** entre las reglas.
- Se considera una función $F_b(N, p)$ que devuelve un número aleatorio obtenido a partir de la distribución binomial $B(N, p)$.
- $R_{sel} \leftarrow \emptyset$

Fase de selección del BBB

for todos los bloques, de acuerdo con el orden considerado, **do**

Sean $r_1, \ldots, r_k$ las reglas del bloque considerado

Sean $p_{r_1}, \ldots, p_{r_k}$ las probabilidades de las reglas del bloque

$N \leftarrow$ el mayor número de veces tal que cualquier regla del bloque es aplicable a C_t

$d \leftarrow 1$

for $i = 1$ to $k - 1$, de acuerdo con el orden seleccionado, **do**

$p_{r_i} \leftarrow \frac{p_{r_i}}{d}$

$n_{r_i} \leftarrow F(N, p_{r_i})$

$N \leftarrow N - n_{r_i}$

$\alpha \leftarrow 1 - p_{r_i}$

$d \leftarrow d * \alpha$

$R_{sel} \leftarrow R_{sel} \cup \{\langle r_i, n_{r_i} \rangle\}$

end for

$n_{r_k} \leftarrow N$

$R_{sel} \leftarrow R_{sel} \cup \{\langle r_k, n_{r_k} \rangle\}$

end for

Fase de ejecución del BBB

for each $< r, n > \in R_{sel}$ **do**

$C_t \leftarrow C_t - n \cdot I(r)$

$C_t \leftarrow C_t + n \cdot r(r)$

Actualizar las cargas eléctricas de C_t de acuerdo con $r(r)$

end for

BBB

Algoritmo de simulación de los sistemas PDP: útil en algunos casos.

Algunas desventajas:

- ⋆ Necesita clasificar todas las reglas del sistema de acuerdo a la LHS.
- ⋆ No contempla la competición de distintas reglas por un mismo objeto.
- ⋆ No chequea la consistencia de cargas en la selección de reglas.

Direct Non-deterministic Distribution algorithm with Probabilities (DNDP)

Definiciones y notaciones:

- ⋆ Se dice que dos reglas r y r' con la misma LHS son **consistentes** si $r(r)$ y $r(r')$ tienen la *misma* carga eléctrica.
- ⋆ $p_{r,j}(t)$ indica la **probabilidad** asociada a la regla r del sistema P situado en el entorno e_j en el instante t.

El algoritmo DNDP fue introducido en el siguiente artículo:

M.A. Martínez, I. Pérez-Hurtado, M.J. Pérez-Jiménez, A. Riscos, M.A. Colomer. A new simulation algorithm for multienvironment probabilistic P systems. In K. Li, Z. Tang, R. Li, A.K. Nagar, R. Thamburaj (eds.) *Proceedings 2010 IEEE Fifth International Conference on Bio-inpired Computing: Theories and Applications*, IEEE Press, Volume 1, September 23-26, 2010, Changsha, China, pp. 59-68.

Esquema algorítmico del DNDP

Input: Un sistema PDP de orden (m, q), con T unidades de tiempo y un número natural $K \geq 1$ (su configuración inicial se notará por C_0).

for $t \leftarrow 0$ to T **do**

$C_t \leftarrow$ configuración del sistema en el instante t

$C'_t \leftarrow C_t$

Inicialización

Primera fase de selección: genera un par de multiconjuntos *consistentes* de reglas aplicables

Segunda fase de selección: genera un multiconjunto *consistente y maximal* de reglas aplicables

Ejecución de las reglas seleccionadas

$C_{t+1} \leftarrow C'_t$

end for

Inicialización del DNDP

$R_\Pi \leftarrow$ conjunto ordenado de reglas de Π

for $j \leftarrow 1$ to m **do**

$R_{E,j} \leftarrow$ conjunto ordenado de reglas de R_E relativo al entorno e_j

$A_j \leftarrow$ conjunto ordenado de reglas de $R_{E,j}$ con probabilidad mayor que 0 en el instante t

$M_j \leftarrow$ conjunto ordenado de pares $\langle label, charge \rangle$, para toda membrana de C_t contenida en el entorno e_j

$B_j \leftarrow \emptyset$

for each $\langle h, \alpha \rangle \in M_j$ (siguiendo el orden considerado) **do**

$B_j \leftarrow B_j \cup$ conjunto ordenado de reglas $u[v]_h^\alpha \leftarrow u'[v']_h^\beta$ de R_Π con probabilidad > 0 en t para e_j

end for

end for

Primera fase de selección del DNDP: consistencia

for $j \leftarrow 1$ to m **do**
  $R^1_{sel,j} \leftarrow \emptyset$
  $R^2_{sel,j} \leftarrow \emptyset$
  for $k \leftarrow 1$ to K **do**
    $D_j \leftarrow A_j \cup B_j$ con un *orden aleatorio*
    for each $r \in D_j$ (siguiendo el orden considerado) **do**
      if r es *consistente* con las reglas en $R^1_{sel,j}$ **then**
        $N' \leftarrow$ máx{número de veces que r es aplicable a C'_t}
        if $N' > 0$ **then**
          if $p_{r,j}(t) = 1$ **then**
            $n \leftarrow F_b(N', 0.5)$
          else
            $N \leftarrow$ máx{número de veces que r es aplicable a C_t}
            $n \leftarrow F_b(N, p_{r,j}(t))$
            if $n > N'$ **then**
              $n \leftarrow N'$
            end if
          end if
          if $n > 0$ **then**
            $C'_t \leftarrow C'_t - n \cdot I(r)$
            $R^1_{sel,j} \leftarrow R^1_{sel,j} \cup \{< r, n >\}$
          else
            $R^2_{sel,j} \leftarrow R^2_{sel,j} \cup \{< r, n >\}$
          end if
        end if
      end if
    end for
  end for
end for

Segunda fase de selección del DNDP: maximalidad

for $j \leftarrow 1$ to m **do**

  $R_{sel,j} \leftarrow R^1_{sel,j} + R^2_{sel,j}$ *orden decreciente en función de las probabilidades de las reglas*

  for each $< r, n > \in R_{sel,j}$ (siguiendo el orden seleccionado) **do**

    if $n > 0 \vee (r$ es *consistente* con las reglas de $R^1_{sel,j})$ **then**

      $N' \leftarrow$ máx{número de veces que r es aplicable a C'_t}

      if $N' > 0$ **then**

        $R^1_{sel,j} \leftarrow R^1_{sel,j} \cup \{< r, N' >\}$

        $C'_t \leftarrow C'_t - N' \cdot I(r)$

      end if

    end if

  end for

end for

Fase de ejecución del DNDP

for each $< r, n > \in R^1_{sel,j}$ **do**

$C'_t \leftarrow C'_t + n \cdot r(r)$

Actualizar las cargas eléctricas de C'_t de acuerdo con $r(r)$

end for

Direct distribution based on Consistent Blocks Algorithmic (DCBA)

Desventaja algoritmo **DNDP**: distorsión en la forma en que las reglas son seleccionadas.

* En vez de seleccionar reglas según sus probabilidades de manera uniforme, este proceso de selección sesga aquellas con menor probabilidad.

Nuevo algoritmo: **D**irect distribution based on **C**onsistent **B**locks **A**lgorithm[6].

* Algoritmo **DCBA**: realiza una distribución uniforme de los recursos a aquellos "bloques" que compiten por ellos.

* Hace uso de un nuevo concepto: **bloque consistente**.

[6] M.A. Martínez, I. Pérez, M. García–Quismondo, L.F. Macías, L. Valencia, A. Romero, C. Graciani, A. Riscos, M.A. Colomer, M.J. Pérez-Jiménez. DCBA: Simulating population dynamics P systems with proportional objects distribution. *Lecture Notes in Computer Science*, 7762 (2013), 257-276

DCBA

Parte izquierda y parte derecha de una regla:

(a) Dada una regla $r \in \mathcal{R}_E$ de la forma $(x)_{e_j} \longrightarrow (y_1)_{e_{j_1}} \cdots (y_h)_{e_{j_h}}$ donde $e_j \in V$ y $x, y_1, \ldots, y_h \in \Sigma$:

- La parte izquierda de r es $LHS(r) = (e_j, x)$.
- La parte derecha de r es $RHS(r) = (e_{j_1}, y_1) \cdots (e_{j_h}, y_h)$.

(b) Dada una regla $r \in \mathcal{R}$ de la forma $u[v]_i^{\alpha} \rightarrow u'[v']_i^{\alpha'}$ donde $1 \leq i \leq q$, $\alpha, \alpha' \in \{0, +, -\}$ y $u, v, u', v' \in \Gamma^*$:

- La parte izquierda de r es $LHS(r) = (i, \alpha, u, v)$. La carga de $LHS(r)$ es $charge(LHS(r)) = \alpha$.
- La parte derecha de r es $RHS(r) = (i, \alpha', u', v')$. La carga de $RHS(r)$ es $charge(RHS(r)) = \alpha'$.

La carga de $LHS(r)$ es la segunda componente de la tupla (idem para $RHS(r)$).

Dados $x \in \Gamma$, $l \in H$, y $r \in \mathcal{R}$ tales que $LHS(r) = (i, \alpha, u, v)$, diremos que (x, l) *aparece* en $LHS(r)$ con multiplicidad k en cualquiera de los siguientes casos:

* $l = i$, y x aparece en el multiconjunto v con multiplicidad k.
* l es la etiqueta de la membrana padre de i, y x aparece en el multiconjunto u con multiplicidad k.

DCBA

Bloque asociado a (i, α, u, v):

* $B_{i,\alpha,u,v} = \{r \in \mathcal{R} : LHS(r) = (i, \alpha, u, v)\}$.

Bloque asociado a (e_j, x):

* $B_{e_j,x} = \{r \in \mathcal{R}_E : LHS(r) = (e_j, x)\}$.

Dos **reglas**, $r_1 \equiv u_1[v_1]_{i_1}^{\alpha_1} \rightarrow u_1'[v_1']_{i_1}^{\alpha_1'}$ y $r_2 \equiv u_2[v_2]_{i_2}^{\alpha_2} \rightarrow u_2'[v_2']_{i_2}^{\alpha_2'}$, son **consistentes** si y solo si $(i_1 = i_2 \wedge \alpha_1 = \alpha_2 \rightarrow \alpha_1' = \alpha_2')$

Un **conjunto de reglas** es **consistente** si todo par de reglas del conjunto es consistente.

El **bloque** $B_{i,\alpha,u,v}$ es **consistente** si y solo si existe α' tal que, para cada $r \in B_{i,\alpha,u,v}$, $charge(RHS(r)) = \alpha'$.

Bloques mutuamente consistentes.

Esquema algorítmico del DCBA

Input: Un sistema PDP de orden (m, q), con T unidades de tiempo y un número natural $A \geq 1$ (**precisión**). Su configuración inicial se notará por C_0.

Inicialización

for $t \leftarrow 1$ to T **do**

Calcular funciones de probabilidad $f_{r,j}(t)$ y $p_r(t)$.

$C'_t \leftarrow C_{t-1}$

Selección de reglas:

- *Fase 1*: distribución
- *Fase 2*: maximalidad
- *Fase 3*: probabilidades

Ejecución de reglas.

$C_t \leftarrow C'_t$

end for

Inicialización del DCBA

Inicialización: se construye una **tabla estática** $\mathcal{T}$ (para cada entorno j, $\mathcal{T}_j$)

- ⋆ Se disponen los bloques en las columnas, y los objetos del alfabeto asociado a cada región del sistema en las filas.
- ⋆ Asocia bloques y objetos por región según las partes izquierdas.
- ⋆ Esta información se complementa con los bloques asociados a las reglas de comunicación entre entornos.
- ⋆ Cada columna de $\mathcal{T}_j$ contiene la información correspondiente a la parte izquierda del bloque.
- ⋆ Cada fila de $\mathcal{T}_j$ contiene la información relacionada con la competición por objetos: dado un objeto, la fila indica cuáles son los bloques que lo requieren.

Inicialización del DCBA

Construcción de la tabla de *distribución estática* $\mathcal{T}$:

- ⋆ Etiquetas de columna: bloques $B_{i,\alpha,\alpha',u,v}$ de $\mathcal{R}$.
- ⋆ Etiquetas de fila: pares (x, i), para todo objeto $x \in \Gamma$, y $0 \leq i \leq q$.
- ⋆ En cada celda de la tabla poner $\frac{1}{k}$ si el objeto de la fila aparece en la LHS del bloque de la columna con multiplicidad k.

for $j = 1$ to m **do** (Construir las tablas *estáticas expandidas* $\mathcal{T}_j$)

$\mathcal{T}_j \leftarrow \mathcal{T}$ (Inicializar la tabla con la original $\mathcal{T}$)

Añadir una fila para cada bloque $B_{e_j,x}$ de $\mathcal{R}_E$, con el valor 1 en la fila $(x, 0)$.

Inicializar los multiconjuntos $B^j_{sel} \leftarrow \emptyset$ y $R^j_{sel} \leftarrow \emptyset$

end for

Fase de selección 1 del DCBA: distribución

La fase de selección 1 utiliza tres procedimientos de filtros auxiliares.

- ⋆ El FILTRO 1 descarta las columnas de la tabla correspondientes a bloques no aplicables por discrepancia de cargas en la parte izquierda y la configuración C'_t.
- ⋆ El FILTRO 2 descarta las columnas con objetos en la parte izquierda que no aparecen en C'_t.
- ⋆ El FILTRO 3 descarta filas vacías.

Estos tres filtros son aplicados al comienzo de la Fase 1, y el resultado es una *tabla dinámica* $\mathcal{T}_j^t$.

Fase de selección 1 del DCBA: distribución

for $j = 1$ to m **do**
  Aplicar filtros a la tabla $\mathcal{T}_j$, usando C'_t y obteniendo $\mathcal{T}_j^t$:
    $\mathcal{T}_j^t \leftarrow \mathcal{T}_j$
    Filtro 1 $(\mathcal{T}_j^t, C'_t)$.
    Filtro 2 $(\mathcal{T}_j^t, C'_t)$.
    Comprobar la *consistencia mutua* para los bloques restantes en $\mathcal{T}_j^t$
      si existe al menos una inconsistencia **entonces**
        reportar el error, y acabar la ejecución
    Filtro 3 $(\mathcal{T}_j^t, C'_t)$
  $a \leftarrow 1$
  until $(a > A) \vee$ (*Todos los mínimos del paso (*) son* 0) **repeat**
    for all file X in $\mathcal{T}_j^t$ **do**
      $RowSum_{X,t,j} \leftarrow$ suma total de valores no nulos en la fila X.
    end for
    $\mathcal{TV}_j^t \leftarrow \mathcal{T}_j^t$ (Una copia temporal)
    for all posición no nula (X, Y) en $\mathcal{T}_j^t$ **do**
      $mult_{X,t,j} \leftarrow$ multiplicidad en C'_t y e_j del objeto en fila X.
      $\mathcal{TV}_j^t(X, Y) \leftarrow \lfloor mult_{X,t,j} \cdot \frac{(\mathcal{T}_j^t(X,Y))^2}{RowSum_{X,t,j}} \rfloor$
    end for
    for all columna no filtrada, etiquetada por el bloque B, en $\mathcal{T}_j^t$ **do**
    $(*)N_B^a \leftarrow \min_{X \in rows(\mathcal{T}_j^t)}(\mathcal{TV}_j^t(X, B))$ El mínimo de la columna)
      $C'_t \leftarrow C'_t - LHS(B) \cdot N_B^a$ (Eliminar el LHS del bloque)
      $B_{sel}^j \leftarrow B_{sel}^j + \{B^{N_B^a}\}$ (Acumular el valor al total)
    end for
    Filtro 2 $(\mathcal{T}_j^t, C'_t)$
    Filtro 3 $(\mathcal{T}_j^t, C'_t)$
    $a \leftarrow a + 1$
end for

Fase de selección 2 del DCBA: maximalidad

for $j = 1$ to m **do**

Elegir un orden aleatorio de los bloques aplicables a los objetos restantes de C'_t

for all bloque B, siguiendo el orden impuesto **do**

$N_B \leftarrow$ número de aplicaciones posibles de B en C'_t.

$B^j_{sel} \leftarrow B^j_{sel} + \{B^{N_B}\}$ (Acumular el valor al total)

$C'_t \leftarrow C'_t - LHS(B) \cdot N_B$ (Eliminar la LHS del bloque B, N_B veces)

end for

end for

Fase de selección 3 del DCBA: probabilidad

for $j = 1$ to m **do** (Para cada entorno e_j)

  for all bloque $B \in B^j_{sel}$ **do**

    $\{r_1, \ldots, r_\alpha\} \leftarrow$ reglas de B con probabilidades $\{p_1, \ldots, p_\alpha\}$

    $\{n_1, \ldots, n_\alpha\} \leftarrow$ Multinomial $M(N_B; p_1, \ldots, p_\alpha)$

    for $k = 1$ to α **do**

    $R^j_{sel} \leftarrow R^j_{sel} + \{r_k^{n_k}\}$

    end for

  end for

  $B^j_{sel} \leftarrow \emptyset$ (Útil para la siguiente iteración)

end for

Fase de ejecución del DCBA

for $j = 1$ to m

for all regla $r^n \in R^j_{sel}$ (Aplicar las RHS de las reglas)

$C'_t \leftarrow C'_t + n \cdot RHS(r)$

Actualizar las cargas eléctricas de C'_t con $RHS(r)$.

end for

$R^j_{sel} \leftarrow \emptyset$ (Útil para la siguiente iteración)

end for

Software para los PDP systems

Hasta el momento se han desarrollado los siguientes simuladores del algoritmo DNDP:

- Implementación en pLingua Core (Java).
  - Versión secuencial.
  - Versión paralela con hilos de la máquina virtual Java (**un hilo para cada entorno**: la máquina no es lo suficientemente eficiente).
- Implementaciones basadas en C++ (empleando librerías que nos permitan manejar paralelismo real en plataformas paralelas).

PGP: Algoritmo de simulación (sin competición de objetos)

Símbolo: un objeto o una bandera.

Bloques: $B_{i,f,u} \equiv (LHS(r) = \{f\}[u]_i)$ and $B_{i,f,u,f}$ $(LHS(r) = \{f\}[u,f]_i)$

Inputs: Un PGP system y $T > 0$ unidades de tiempo

for $t \leftarrow 1$ to T **do**

Fase de selección:

Chequea las **banderas** de los bloques.
Distribuir los símbolos disponibles entre los bloques.
Consumir los símbolos.
Distribuir las aplicaciones de las reglas de cada bloque (binomial con parámetros: la probabilidad de las reglas y el máximo número de aplicaciones del bloque).

Fase de ejecución:

Producir símbolos.

Está inspirado en el algoritmo DCBA[7]

[7] M.A. Martínez, I. Pérez, M. García–Quismondo, L.F. Macías, L. Valencia, A. Romero, C. Graciani, A. Riscos, M.A. Colomer, M.J. Pérez. DCBA: Simulating population dynamics P systems with proportional objects distribution. *Lecture Notes in Computer Science*, 7762 (2013), 257-276

Software para los PGP systems

Hasta el momento se han desarrollado los siguientes simuladores del algoritmo antes descrito:

* Implementación en pLingua Core (Java).
* Implementaciones basadas en C++ (empleando librerías que nos permitan manejar paralelismo real en plataformas paralelas).
* Implementaciones basadas en CUDA/C++ (con la ventaja de usar la arquitectura paralela de las tarjetas gráficas GPUs).

Integración of PGP systems in P–Lingua.