

Computación Bio–inspirada

Anexo del tema IX: Una solución eficiente del problema SAT mediante sistemas P con membranas activas

Mario de J. Pérez Jiménez

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

marper@us.es

<http://www.cs.us.es/~marper/>

Máster Universitario en Lógica Computación e Inteligencia Artificial
Curso 2019-2020

Diseño de una solución eficiente de SAT con una familia de sistemas P de \mathcal{AM}

Para cada $(m, n) \in \mathbf{N}^2$: $\langle m, n \rangle = \frac{(m+n)(m+n+1)}{2} + m$.

Sea $\Pi(\langle m, n \rangle) = (\Gamma, \Sigma, L, \mu, \mathcal{M}_1, \dots, \mathcal{M}_q, \mathcal{R}, i_{in})$, en donde:

- $\Sigma = \{x_{i,j}, \bar{x}_{i,j} : 1 \leq i \leq m, 1 \leq j \leq n\}$.
- $\Gamma = \Sigma \cup \{c_k : 1 \leq k \leq m+2\} \cup \{d_k : 1 \leq k \leq 3n+2m+3\} \cup \{r_{i,k} : 0 \leq i \leq m, 1 \leq k \leq 2n\} \cup \{e, t\} \cup \{\text{Yes}, \text{No}\}$.
- $L = \{1, 2\}$.
- $\mu = [[]_2]_1$.
- $\mathcal{M}_1 = \emptyset$ y $\mathcal{M}_2 = \{d_1\}$.
- $i_{in} = 2$.

El conjunto de reglas \mathcal{R} es el siguiente:

- (a) $\{[2d_k]_2^0 \rightarrow [2d_k]_2^+ [2d_k]_2^- : 1 \leq k \leq n\}$.
- (b) $\{[2x_{i,1} \rightarrow r_{i,1}]_2^+, [2\bar{x}_{i,1} \rightarrow r_{i,1}]_2^-, [2x_{i,1} \rightarrow \lambda]_2^-, [2\bar{x}_{i,1} \rightarrow \lambda]_2^+ : 1 \leq i \leq m\}$.
- (c) $\{[2x_{i,j} \rightarrow x_{i,j-1}]_2^+, [2x_{i,j} \rightarrow x_{i,j-1}]_2^- : 1 \leq i \leq m, 2 \leq j \leq n\}$.
 $\{[2\bar{x}_{i,j} \rightarrow \bar{x}_{i,j-1}]_2^+, [2\bar{x}_{i,j} \rightarrow \bar{x}_{i,j-1}]_2^- : 1 \leq i \leq m, 2 \leq j \leq n\}$.
- (d) $\{[2d_k]_2^+ \rightarrow [2]_2^0 d_k, [2d_k]_2^- \rightarrow [2]_2^0 d_k : 1 \leq k \leq n\}$.
 $\{d_k [2]_2^0 \rightarrow [2d_{k+1}]_2^0 : 1 \leq k \leq n-1\}$.
- (e) $\{[2r_{i,k} \rightarrow r_{i,k+1}]_2^0 : 1 \leq i \leq m, 1 \leq k \leq 2n-1\}$.
- (f) $\{[1d_k \rightarrow d_{k+1}]_1^0 : n \leq k \leq 3n-3\}; [1d_{3n-2} \rightarrow d_{3n-1}e]_1^0$.
- (g) $e[2]_2^0 \rightarrow [2c_1]_2^+; [1d_{3n-1} \rightarrow d_{3n}]_1^0$.
- (h) $\{[1d_k \rightarrow d_{k+1}]_1^0 : 3n \leq k \leq 3n+2m+2\}$.
- (i) $[2r_{1,2n}]_2^+ \rightarrow [2]_2^- r_{1,2n}$.
- (j) $\{[2r_{i,2n} \rightarrow r_{i-1,2n}]_2^- : 1 \leq i \leq m\}$.
- (k) $r_{1,2n}[2]_2^- \rightarrow [2r_{0,2n}]_2^+$.
- (l) $\{[2c_k \rightarrow c_{k+1}]_2^- : 1 \leq k \leq m\}$.
- (m) $[2c_{m+1}]_2^+ \rightarrow [2]_2^+ c_{m+1}$.
- (n) $[1c_{m+1} \rightarrow c_{m+2}t]_1^0$.
- (o) $[1t]_1^0 \rightarrow [1]_1^+ t$.
- (p) $[1c_{m+2}]_1^+ \rightarrow [1]_1^- \text{Yes}$.
- (q) $[1d_{3n+2m+3}]_1^0 \rightarrow [1]_1^+ \text{No}$.

La familia Π proporciona una solución lineal del problema SAT

Consideramos la **codificación polinomial** (cod, s) de I_{SAT} en la familia Π .

- Para cada fórmula $\varphi = C_1 \wedge \dots \wedge C_m$ en FNC con $Var(\varphi) = \{x_1, \dots, x_n\}$:
 - ★ $s(\varphi) = \langle n, m \rangle$.
 - ★ $cod(\varphi) = \{x_{i,j} : x_j \in C_i\} \cup \{\bar{x}_{i,j} : \neg x_j \in C_i\}$

La función s es primitiva recursiva, computable en tiempo polinomial y biyectiva de \mathbf{N}^2 en \mathbf{N} .

La función cod es computable en tiempo polinomial y para cada $\varphi \in I_{SAT}$, se tiene que $cod(\varphi)$ es un multiconjunto de entrada del sistema $\Pi(s(\varphi))$.

Si \mathcal{C} es la computación del sistema $\Pi(s(\varphi))$ con entrada $cod(\varphi)$, notaremos por \mathcal{C}^i la configuración obtenida tras ejecutar i pasos de \mathcal{C} .

La ejecución del sistema $\Pi(s(\varphi))$ con entrada $\text{cod}(\varphi)$ se estructura en cuatro etapas:

- La etapa de **generación** halla todas las posibles valoraciones (en las membranas internas).
 - ★ La presencia de d_i ($1 \leq i \leq n$) en la piel indica que se han generado todas las posibles valoraciones de $\{x_1, \dots, x_i\}$.
 - ★ Simultáneamente, en cada membrana interna aparecerán todas las cláusulas verdaderas por la valoración parcial que codifica esa membrana (mediante los objetos $r_{i,k}$).
 - ★ Finaliza cuando el objeto d_n aparece en la piel.
- La etapa de **sincronización** pone los segundos índices de $r_{i,k}$ iguales a $2n$.
 - ★ En cada paso de esta etapa, d_i ($n \leq i \leq 3n - 1$) evoluciona a d_{i+1} .
 - ★ Finaliza cuando los objetos d_{3n} aparecen en la piel (momento en que cada membrana interna tiene una carga $+$ y contiene c_1).
- La etapa de **chequeo** determina el número de cláusulas verdaderas que hay en cada membrana interna.
 - ★ La presencia de c_i en una membrana interna prueba que las cláusulas C_1, \dots, C_{i-1} son verdaderas por la valoración codificada.
 - ★ Para cada c_i (with $1 \leq i \leq m$), el objeto c_{i+1} aparece en *alguna* membrana tras ejecutar 2 pasos.
 - ★ Finaliza cuando el objeto d_{3n+2m} aparece en la piel.
- La etapa de **salida** produce la respuesta del sistema.
 - ★ La salida *Yes* se produce si aparece el objeto c_{m+1} en alguna membrana interna de $\mathcal{C}^{5n+2m-1}$. Entonces, el sistema devuelve *Yes* tras 4 pasos.
 - ★ La salida *No* se produce si ningún objeto c_{m+1} aparece en ninguna membrana interna de $\mathcal{C}^{5n+2m-1}$. Entonces, después de 4 pasos el sistema devuelve *No*.

- Π es polinomialmente uniforme por MT pues las reglas están descritas recursivamente y la cantidad de recursos usados para su construcción es:
 - ★ Número total de **objetos**: $4mn + 3m + 5n + 9 \in \Theta(m \cdot n)$
 - ★ Número de **membranas iniciales** 2.
 - ★ Máximo **cardinal** de los **multiconjuntos iniciales**: 1.
 - ★ Número total de **reglas**: $6mn + 3m + 6n + 10 \in \Theta(m \cdot n)$.
 - ★ Máxima **longitud** de una **regla**: 9.
- Π es polinomialmente acotada con respecto a (cod, s) : el tiempo de ejecución de $\Pi(s(\varphi)) + cod(\varphi)$ es:
 - ★ $3n - 1$ pasos en la fase de **generación**.
 - ★ $2n$ pasos en la fase de **sincronización**.
 - ★ $2m$ pasos en la fase de **chequeo**.
 - ★ 4 pasos en la fase de **salida**.

Tiempo total de ejecución: $5n + 2m + 3 \in \Theta(n + m)$.

- Cada sistema de la familia Π es determinista.
- Π es adecuada y completa respecto a $(\text{SAT}, \text{cod}, s)$,
 - ★ Si alguna membrana interna de $\mathcal{C}^{5n+2m-1}$ contiene un objeto c_{m+1} , entonces en el último paso el sistema envía un objeto *Yes* al entorno.
 - ★ Si ninguna membrana interna de $\mathcal{C}^{5n+2m-1}$ contiene un objeto c_{m+1} , entonces en el último paso el sistema envía un objeto *No* al entorno.
 - ★ Si la computación de $\Pi(s(\varphi))$ con entrada $\text{cod}(\varphi)$ es de aceptación, entonces la fórmula φ es satisfactible (**Adecuación**).
 - ★ Si la la fórmula φ es satisfactible, entonces la computación de $\Pi(s(\varphi))$ con entrada $\text{cod}(\varphi)$ es de aceptación (**Compleitud**).

Teorema: $\text{SAT} \in \text{PMC}_{\mathcal{AM}}$.

Corolario: $\text{NP} \cup \text{co-NP} \subseteq \text{PMC}_{\mathcal{AM}}$.

Se tiene que

$$(a) \quad \text{P} = \text{PMC}_{\mathcal{NAM}}.$$

$$(b) \quad \text{NP} \cup \text{co-NP} \subseteq \text{PMC}_{\mathcal{AM}}.$$

Luego en el marco de los sistemas P con membranas activas:

- La **regla de división** proporciona una frontera de la tratabilidad (suponiendo que $\text{P} \neq \text{NP}$).

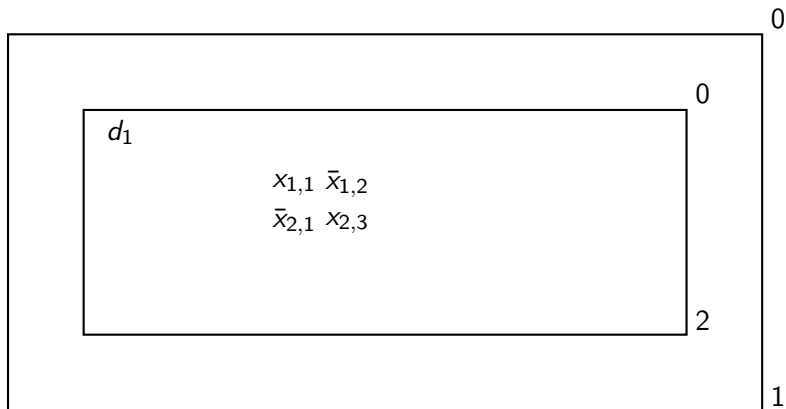
Ejemplo de una instancia concreta

► $\varphi = (x_1 \vee \bar{x}_2) \wedge (\bar{x}_1 \vee x_3) \qquad n = 3 \text{ y } m = 2$

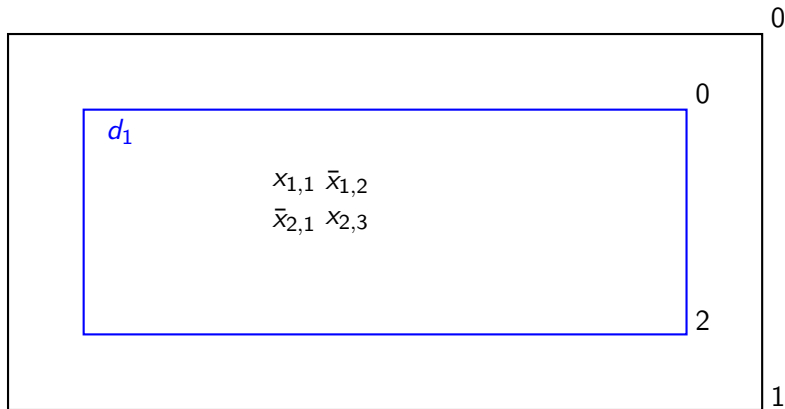
► $s(\varphi) = \langle 2, 3 \rangle = 17$

► $cod(\varphi) = \{x_{1,1}, \bar{x}_{1,2}, \bar{x}_{2,1}, x_{2,3}\}$

Configuración inicial del sistema

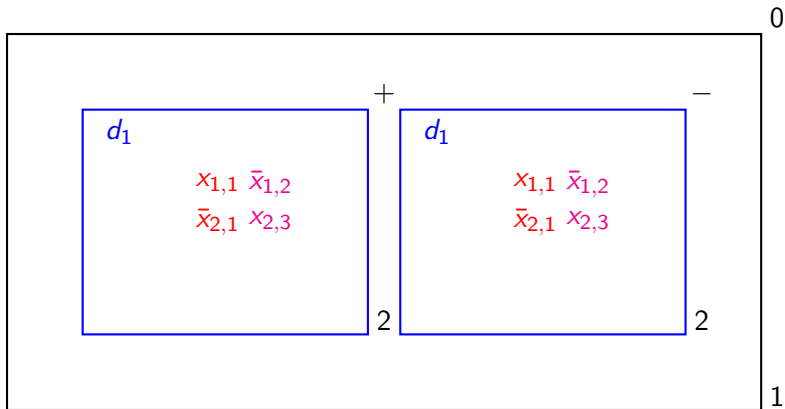


Fase de generación y primer chequeo (I)



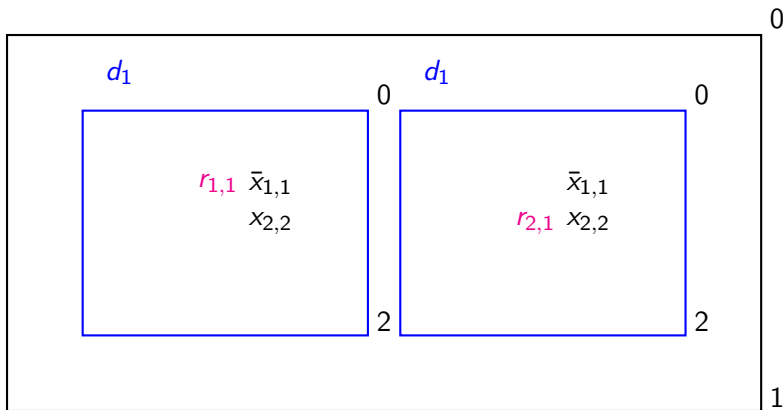
Reglas: (a)

Fase de generación y primer chequeo (II)



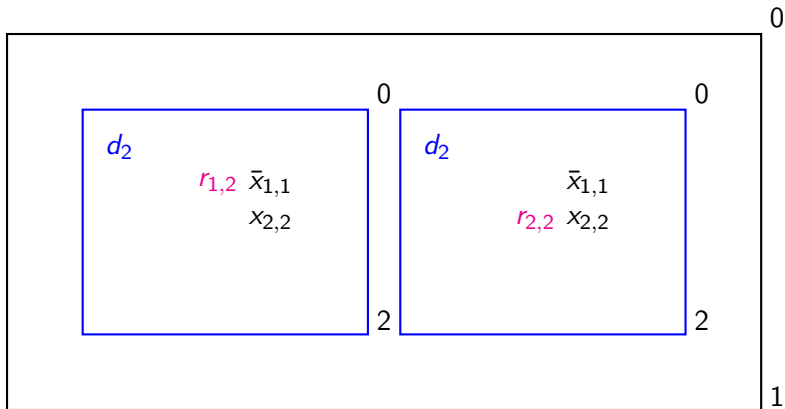
Reglas: (d), (b), (c)

Fase de generación y primer chequeo (III)



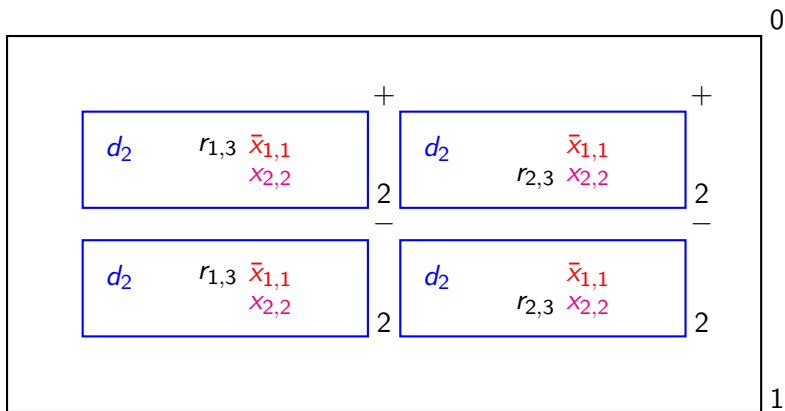
Reglas: (d),(e)

Fase de generación y primer chequeo (IV)



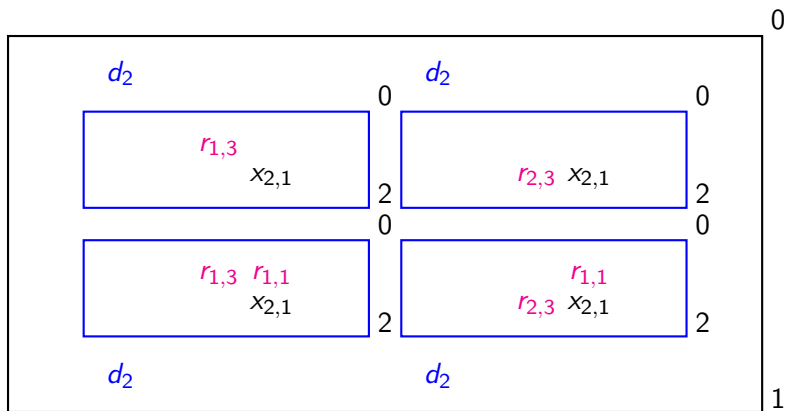
Reglas: (a),(e)

Fase de generación y primer chequeo (V)



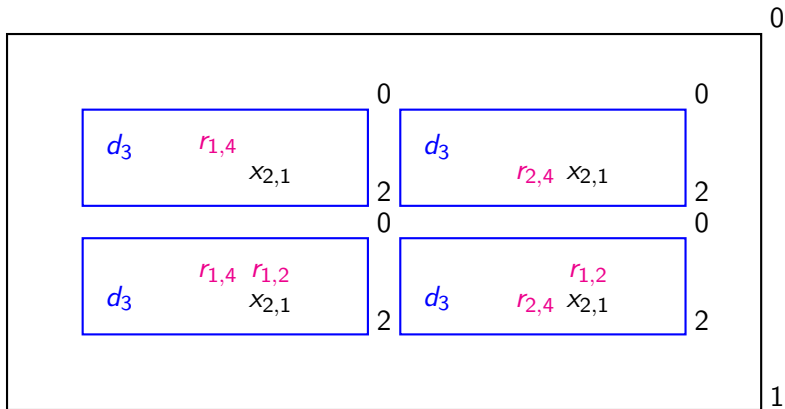
Reglas: (d),(b),(c)

Fase de generación y primer chequeo (VI)



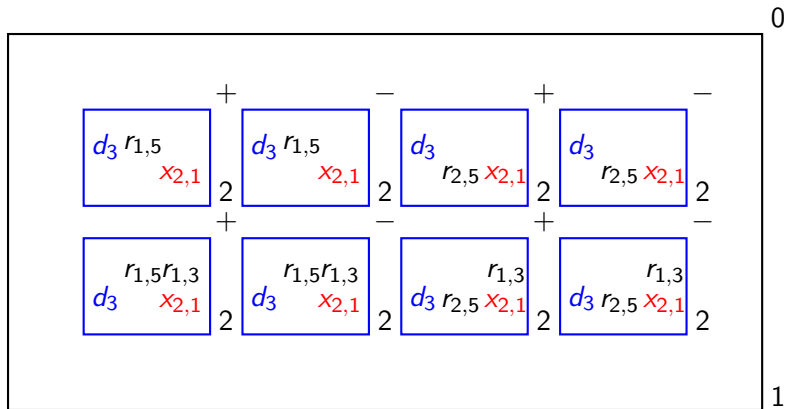
Reglas: (d),(e)

Fase de generación y primer chequeo (VII)

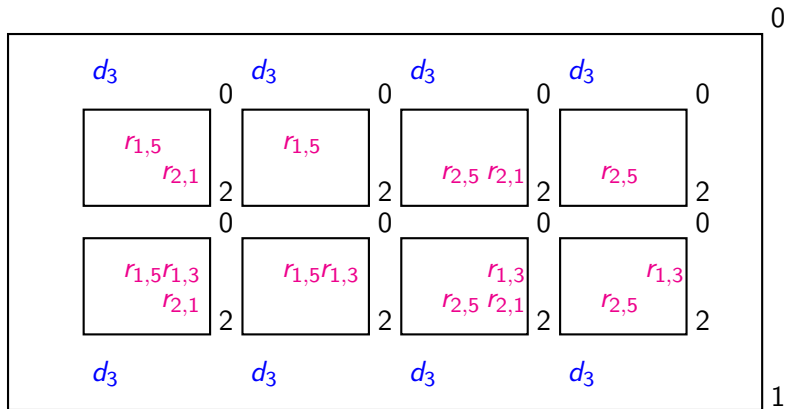


Reglas: (a),(e)

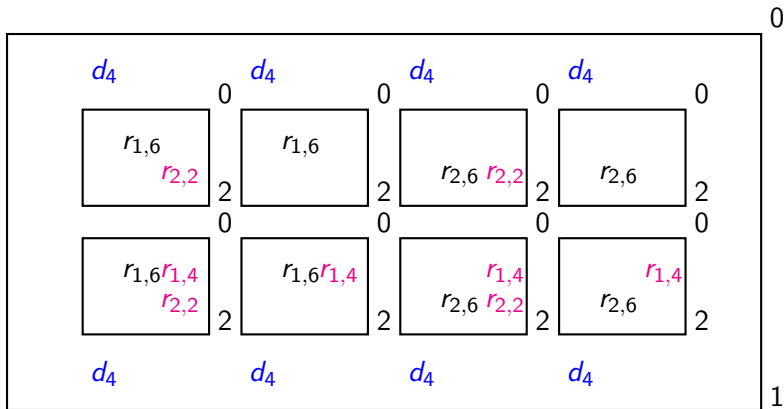
Fase de generación y primer chequeo (VIII)



Fase de sincronización (I)

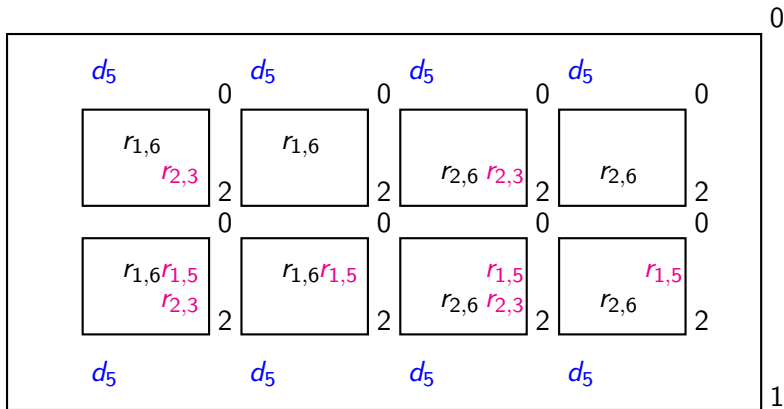


Fase de sincronización (II)



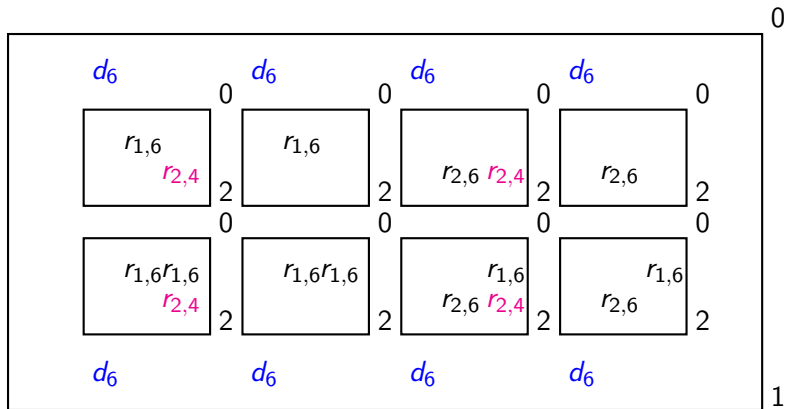
Reglas: (f),(e)

Fase de sincronización (III)



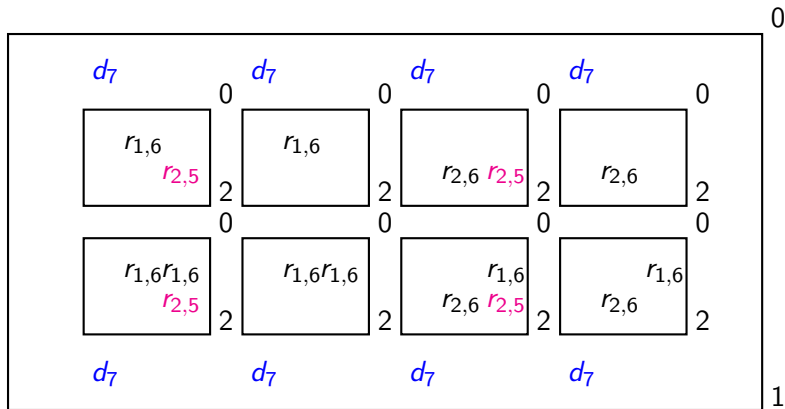
Reglas: (f), (e)

Fase de sincronización (IV)



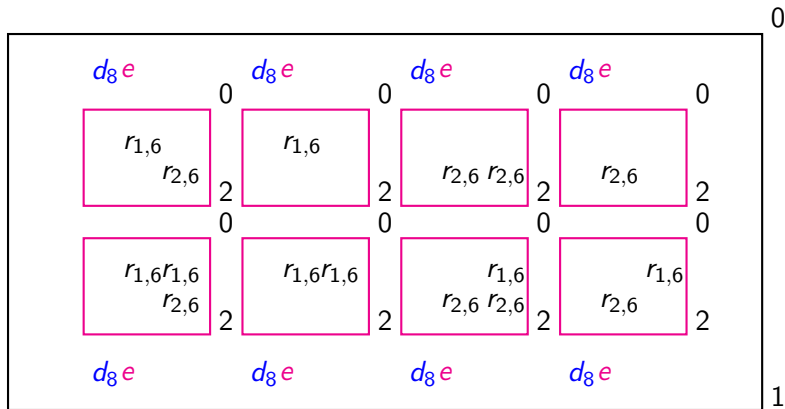
Reglas: (f), (e)

Fase de sincronización (V)



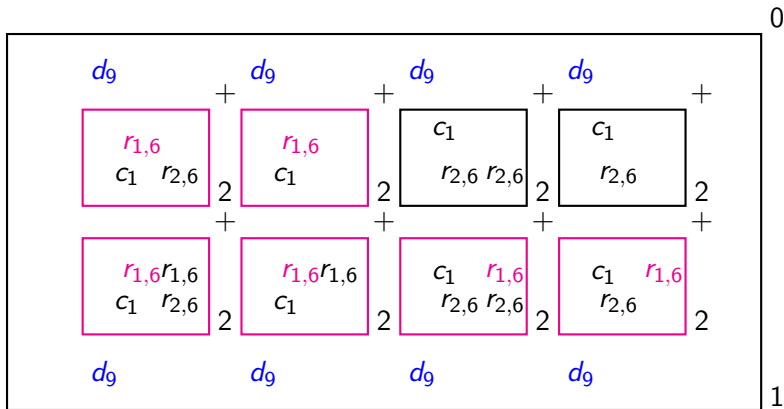
Reglas: (f), (e)

Fase de sincronización (VI)



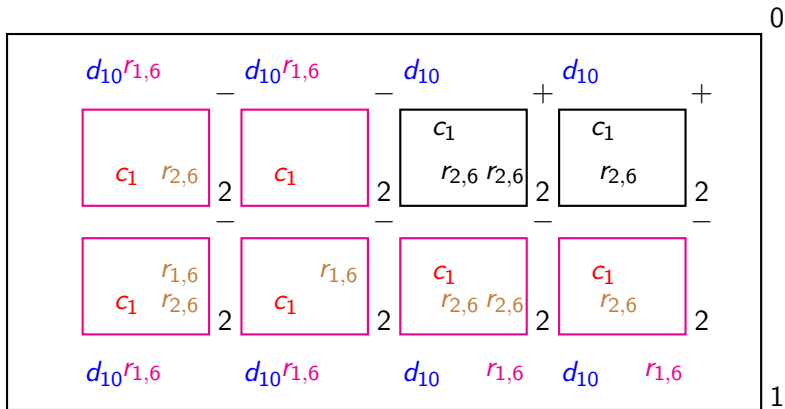
Reglas: (d_8), (e)

Fase de segundo chequeo (I)



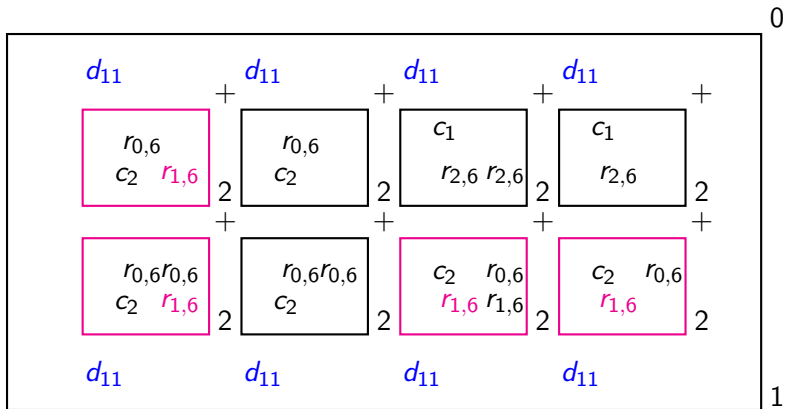
Reglas: (h),(i)

Fase de segundo chequeo (II)



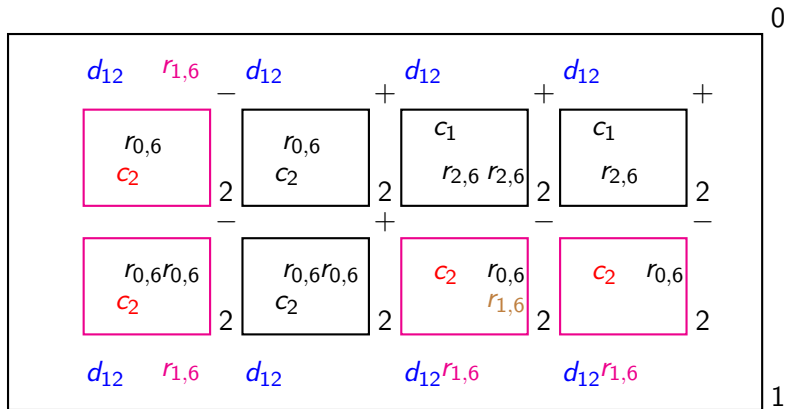
Reglas: (h),(k),(j),(l)

Fase de segundo chequeo (III)



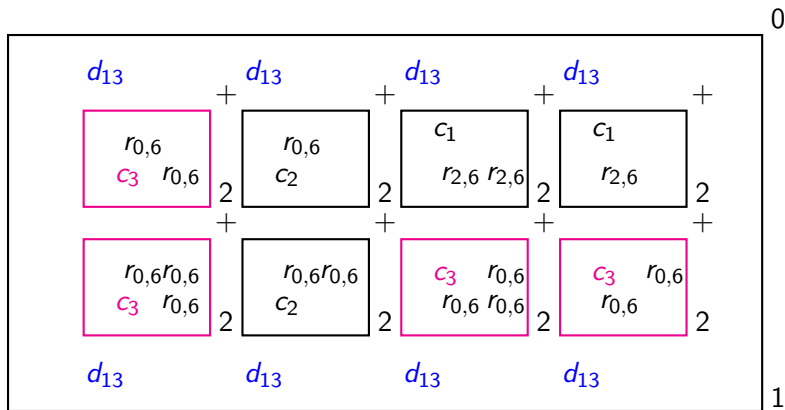
Reglas: (h),(i)

Fase de segundo chequeo (IV)



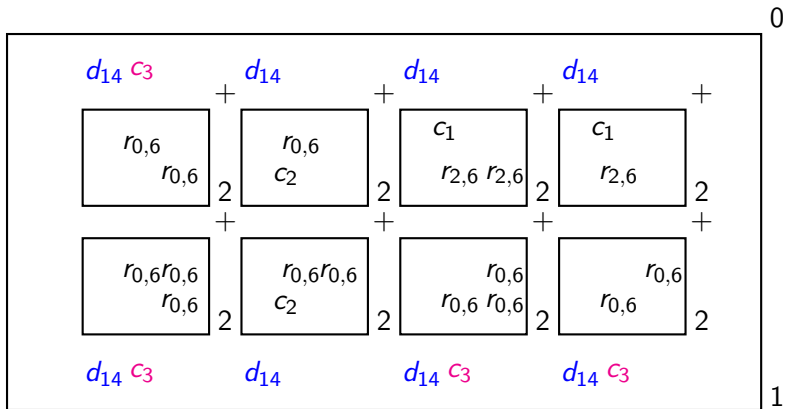
Reglas: (h),(k),(j),(l)

Fase de segundo chequeo (V)



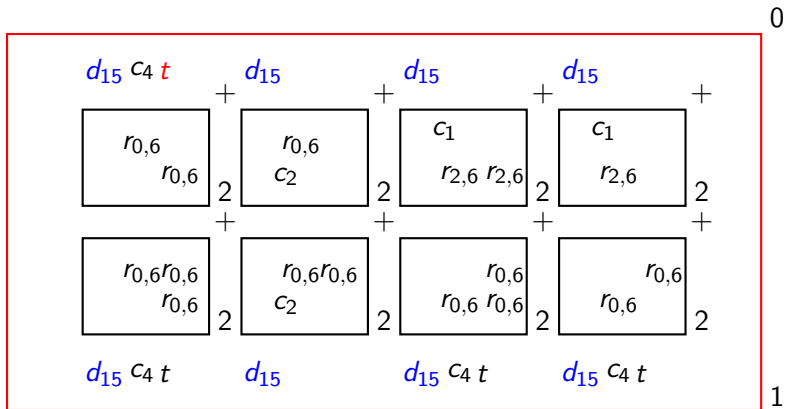
Reglas: (h),(m)

Fase de salida (I)



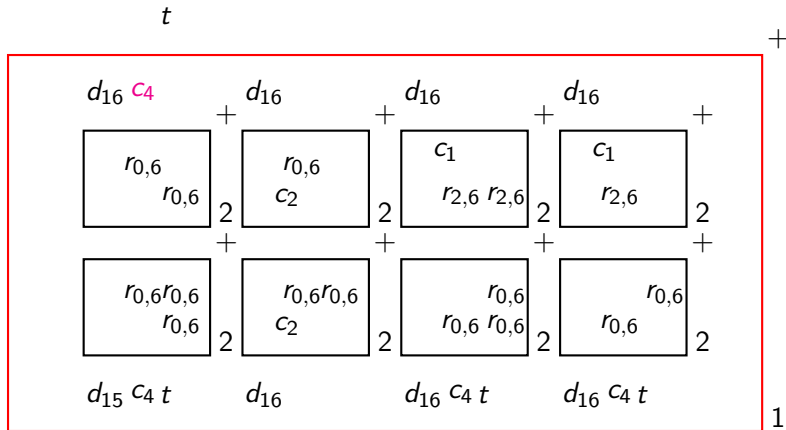
Reglas: (h),(n)

Fase de salida (II)



Reglas: (h),(o)

Fase de salida (III)



Reglas: (p)

Fase de salida (IV)

Yes t

