# Problemas de Satisfacción de Restricciones Temporales no Binarias Mediante Poliedros Dinámicos

Miguel A. Salido, Adriana Giret, Federico Barber
Departamento de Sistemas Informáticos y Computación
Universidad Politécnica de Valencia
Camino de Vera s/n, 46071-Valencia
{msalido, agiret, fbarber}@dsic.upv.es

**Resumen.** *Hoy en día, muchos problemas reales pueden ser modelados como problemas de satisfacción de restricciones no binarias. Las técnicas actuales de transformación de restricciones no binarias a binarias pueden ser muy costosas e incluso impracticables, por lo que en ciertos casos es necesario el uso de técnicas que manejen eficientemente las restricciones no binarias de forma directa. En este artículo presentamos un modelo eficiente que resuelve problemas de satisfacción de restricciones no binarias de forma natural, pudiendo manejar restricciones temporales no binarias (lineales o no lineales) con dominios continuos que se insertan en el problema de forma incremental, obteniendo resultados importantes como la consistencia del problema, las soluciones que el usuario necesite, o aquella solución que optimice una o varias funciones objetivo.*

**Palabras Clave**: Satisfacción de restricciones, heurística, restricciones temporales no binarias.

## 1 Introducción

Hoy en día, muchos problemas reales pueden ser modelados como problemas de satisfacción de restricciones (CSP's) no binarias. Sin embargo la mayoría de los investigadores que trabajan en problemas de satisfacción de restricciones centran su atención en problemas binarios. Es bien conocido que todo problema de satisfacción de restricciones no binarias puede ser transformado a un problema binario equivalente [Rossi'90][Dechter'90].

Sin embargo, esta transformación de restricciones tiene ciertos inconvenientes. En la práctica, esta transformación puede ser poco eficiente debido a su alto coste computacional haciendo que con frecuencia esta transformación sea impracticable [Larrosa'98] [Bessière'99]. Además, el proceso de transformación genera nuevas variables, las cuales pueden tener dominios muy grandes, causando unos requerimientos de memoria extra para los algoritmos que lo resuelven. Por ello, en algunos casos, resolver el problema binarizado puede resultar muy ineficiente [Bacchus'98].

Por lo tanto, a menudo resulta más adecuado manejar las restricciones no binarias de forma directa, de manera que los problemas mantengan sus formulaciones originales. Las técnicas actuales que manejan restricciones no binarias en su forma natural se basan en técnicas tradicionales de CSP's, como *backtraking*, y además se centran en dominios discretos.

En este artículo proponemos un modelo llamado "MAS" (Sistema Multi-Algoritmo) para resolver problemas de satisfacción de restricciones temporales (TCSP) no binarias sobre dominios continuos de forma directa y natural. MAS está formado por varios algoritmos: HSA [Salido'01], OFHH [Salido'01a], NFHH [Salido'01b] y POLYSA [Salido'01c]. Cada uno de estos algoritmos tiene unas características y complejidad específicas. MAS tiene todas las ventajas de cada algoritmo y dependiendo de la tipología del problema ejecutará uno u otro, o incluso una mezcla de ambos. Este modelo lleva a cabo la búsqueda mediante un poliedro que mantiene en sus facetas las restricciones de inegualdad, actualizandose cada vez que una nueva restricción temporal se inserta en el problema.

Este artículo sigue la siguiente estructura. En un principio presentamos la tipología de restricciones a manejar, a continuación proponemos y detallamos el modelo MAS presentando una evaluación con respecto a las técnicas más conocidas, y finalmente exponemos las conclusiones y las propuestas para futuros trabajos.

# 2 Preliminares

Brevemente, el problema de satisfacción de restricciones temporales (TCSP) que nuestro modelo maneja, consiste en:

- Un conjunto de variables $X = \{x_1, ..., x_n\}$;
- Un conjunto de dominios $\{D_1, D_2, ..., D_n\}$ asociado a cada variable en *X*. Cada dominio $D_i$ esta definido como un intervalo continuo de la forma $D_i = [l_i, u_i]$.

- Un conjunto de restricciones estáticas $C=\{c_1,...,c_p\}$ limitando los valores que las variables pueden simultáneamente tomar. Estas restricciones componen las restricciones iniciales del problema.
- Un conjunto de restricciones dinámicas $C'=\{c'_1,...,c'_p\}$ que se van insertando en el sistema de forma incremental.

Una solución a un TCSP es una asignación de un valor de su dominio a cada una de las variables de tal manera que se satisfagan todas las restricciones del problema.

De esta manera el objetivo puede ser:

- obtener una solución, sin preferencia alguna;
- obtener varias, todas o aquellas soluciones que puedan ser mas resistentes;
- obtener los dominios mínimos de cada una de las variables;
- obtener una solución que optimice una o varias funciones objetivo (lineales o no lineales) definida en términos de algunas variables.

## 2.1 Notación y definiciones

A continuación resumiremos las notación que utilizaremos en este artículo.

**General**: El número de variables de un CSP lo denotaremos por *n*. El número de restricciones totales lo denotaremos por *c*. Sin embargo, las restricciones de desigualdad las denotaremos por $c_{\bullet}$ , las restricciones de inegualdad por $c_{\bullet}$ y por último las restricciones no lineales por $c_f$.

**Variables**: Para representar las variables utilizaremos las últimas letras del alfabeto en cursiva, por ejemplo *x*, *y*, *z*, así como esas mismas letras con un subíndice, por ejemplo $x_1$, $x_i$, $x_j$. Estos subíndices son letras seleccionadas por mitad del alfabeto o números enteros.

**Dominios/Valores**: El dominio de una variable $x_i$ lo denotamos por $D_i=[l_i,u_i]$, por lo que la longitud del dominio de la variable $x_i$ es $u_i$-$l_i$. Es importante remarcar que estamos trabajando con restricciones temporales, en los que las variables son puntos de tiempo acotadas en dominios continuos.

**Restricciones**: Sea $X=\{x_1, ..., x_k\}$ un conjunto de variables reales. Sea ♀ un polinomio de grado *n*, es decir $\alpha=\sum_{i=1}^{n} p_i x_i$ , sobre *X* y *b* un número real. Una relación lineal sobre *X* es una expresión de la forma $\alpha r b$ donde $r \in \{<,\leq,=,\neq,\geq,>\}$. Una desigualdad lineal sobre *X* es una expresión de la forma $\alpha \neq b$. Una inegualdad lineal sobre *X* es una expresión de la forma $\alpha \leq b$. Las restricciones temporales no binarias que vamos a manejar son de la forma:

$$\text{Inegualdad lineal } c_\bullet : \sum_{i=1}^{n} p_i x_i \leq b \tag{1}$$

$$\text{Desigualdad lineal } c_\bullet : \sum_{i=1}^{n} p_i x_i \neq b \tag{2}$$

$$\text{Restricciones no lineales } c_f : F(x) \neq b : F \text{ no lineal} \tag{3}$$

donde $x_i$ son variables que están acotadas en intervalos continuos $x_i \in [l_i, u_i]$, $b$ es una constante real y $n \geq 1$.
Mediante la combinación de restricciones del tipo (1) y (2) podemos obtener cualquier relación lineal. De esta manera MAS puede manejar relaciones lineales así como restricciones no lineales del tipo (3).

**Ejemplo** La restricción $x_4 - x_3 \leq 3$ es una restricción binaria, la restricción $2x_1 - x_2 + 3x_3 \leq 2$ es una restricción no binaria. Ambas son inegualdades lineales. La restricción $x_1 + 3x_2 + 5x_5 \neq 1$ es una desigualdad lineal y la restricción $2x_1x_2 - 3x_3 \neq 5$ es una restricción no lineal.

# 3 Especificación de MAS y sus algoritmos

Inicialmente MAS puede ser considerado como un clásico resolutor de problemas de satisfacción de restricciones temporales, donde hay un conjunto estático de restricciones que la solución debe satisfacer. En la figura 1 se presenta el proceso básico de MAS.

- Inicialmente MAS estudia los parámetros significativos tales como el número de variables del problema y el número de restricciones de inegualdad. Dependiendo de estos parámetros MAS selecciona el algoritmo más apropiado, generando un poliedro inicial (paso 1), con *v* vértices aleatorios (*v*= $2^n$ vértices para HSA, $2n$ para OFHH, $2n^2$ para NFHH y $2n^3$ para POLYSA) generados mediante el producto cartesiano de los valores extremos de los dominios de las variables $D_1 \times D_2 \times \cdots \times D_n$, de forma que, en las heurísticas, a cada faceta del poliedro se le asignan $v/2n$ vértices no asignados anteriormente en otra faceta adyacente. La asignación de estos vértices es aleatoria y no influye en el resultado la selección de un conjunto u otro.

- Para cada restricción de entrada $c_\bullet$ , MAS lleva a cabo la prueba de la consistencia (paso 2). Si la restricción no es consistente MAS detecta dicha inconsistencia y finaliza la ejecución. Sin embargo si la restricción es consistente, el algoritmo estudia si la restricción es redundante. Si es así, el poliedro no se actualiza y continua analizando la siguiente restricción. Si no es redundante, se actualiza el poliedro (paso 3) mediante técnicas de programación lineal.
- Una vez analizadas las restricciones estáticas de inegualdad, MAS estudia la consistencia de las restricciones de desigualdad (paso 4).
- Finalmente MAS devuelve al usuario información relevante del problema actual, como la consistencia del problema, las soluciones que el usuario requiera, aquellas soluciones que sean más resistente a los cambios, los dominios mínimos de las variables o aquella solución que optimice una función objetivo o multi-objetivo. Los vértices están etiquetados almacenando su antigüedad en dicha etiqueta, de forma que si las restricciones siguen un comportamiento similar, se puede prever que soluciones pueden satisfacerse con las sucesivas restricciones.

Una vez finalizado el comportamiento estático con las restricciones iniciales, el usuario puede insertar nuevas restricciones *on-line* y MAS estudiará la consistencia del problema sin la necesidad de resolver todo el problema de nuevo. De esta manera, el usuario puede obtener la consistencia del problema de una manera rápida, para más tarde actualizarse el poliedro y dejar el sistema consistente. Así, MAS es capaz de manejar problemas dinámicos cuyas restricciones temporales se van incorporando al sistema de forma incremental y devolviendo en cada momento toda la información que el usuario solicite.

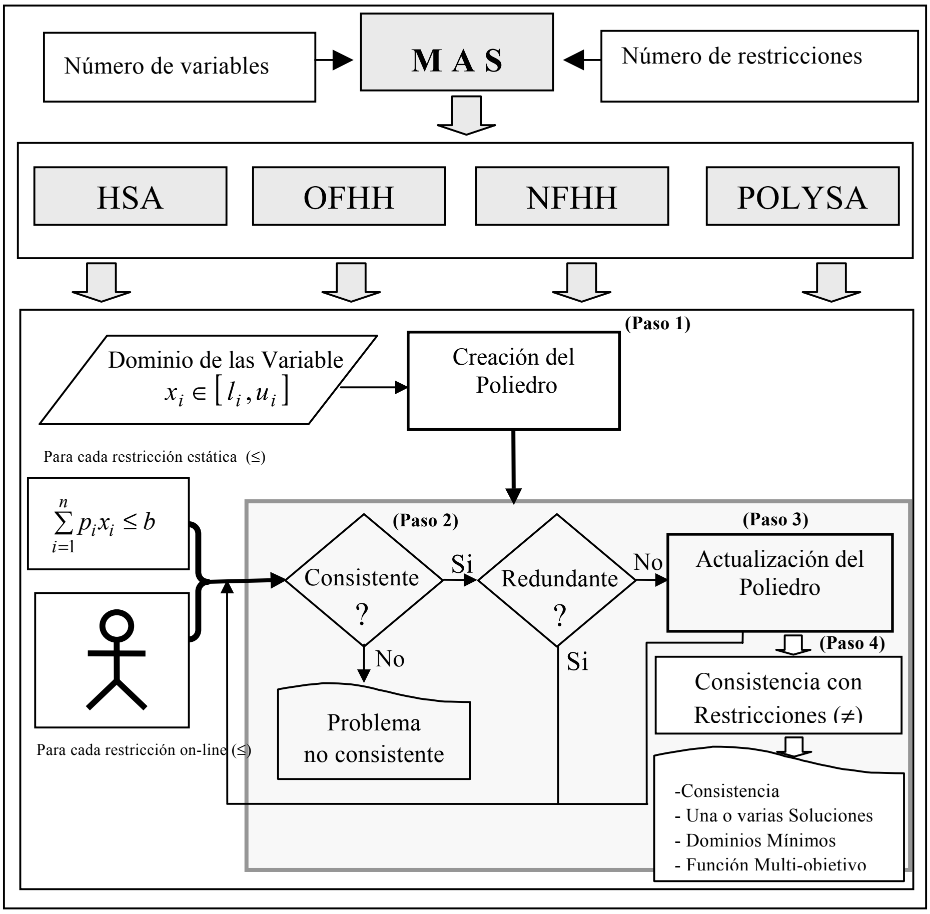


*Figura 1*. Esquema general de MAS.

**Teorema de Completitud (*k,p*).**

Una técnica heurística utilizada por MAS con *2nk* vértices iniciales es completa para $n<p$.

*Demostración*: Supongamos por reducción al absurdo que esta técnica heurística no es completa para $n<p$. El poliedro generado mediante el producto cartesiano de los valores extremos de los dominios de las variables consta de $2n$ facetas y $2^n$ vértices.

Esta técnica genera en cada faceta $k$ vértices no repetidos. Así, esta técnica genera un total de $2nk$ vértices. Si no fuera completa para $n<p$ implicaría que hay vértices que esta técnica no los contempla para $n<p$, por lo tanto $2nk < 2^n$ para $n<p$. Esto es una contradicción, pues $2^n < 2nk$ para $n<p$. Por tanto, la técnica heurística es completa para $n<p$.

**Teorema de Corrección (*k,p*).**

Una técnica heurística utilizada por MAS con $2nk$ vértices iniciales es completa para $n$● $p$.

*Demostración*: Para $n \geq p$, se cumple $2nk < 2^n$ por lo que esta técnica heurística no es completo. Sin embargo cuando el algoritmo clasifica el problema como consistente, podemos garantizar el resultado obtenido puesto que el conjunto de vértices que maneja esta técnica es un subconjunto del conjunto total que maneja el poliedro completo (HSA). De esta forma, cualquier solución obtenida por MAS es correcta para $n \geq p$.

**Algoritmo completo HSA**

HSA [Salido'01] es un algoritmo completo y correcto. Los vértices se generan mediante el producto cartesiano de todos los límites de los dominios de las variables. Es por ello por lo que su complejidad es O($2^n$). Ese algoritmo es apropiado para problemas con pocas variables, muchas restricciones de desigualdad, grandes dominios y pocas restricciones de inegualdad.

**Algoritmo heurístico OFHH**

OFHH [Salido'01a] es el algoritmo heurístico más simple de todos. Al poliedro inicial sólo se le asigna, de forma aleatoria, un único vértice que no haya sido asignado anteriormente, por lo que el poliedro inicial consta de $2n$ vértices ($k=1$). Este algoritmo satisface el teorema de completitud y corrección para $k=1$ y $p=3$. Su complejidad computacional es O($n$) y es un algoritmo apropiado para problemas con muchas variables, muchas restricciones de desigualdad, grandes dominios y pocas restricciones de inegualdad. Este algoritmo falla cuando el número de inegualdades aumenta, por lo que MAS sólo lo utilizará para problemas con pocas restricciones de este tipo.

**Algoritmo heurístico NFHH**

NFHH [Salido'01b] es el segundo algoritmo heurístico más simple de todos. Al poliedro inicial sólo se le asigna, de forma aleatoria, $n$ vértices que no hayan sido asignado anteriormente, por lo que el poliedro inicial consta de $2n^2$ vértices ($k=n$). Este algoritmo satisface el teorema de completitud y corrección para $k=n$ y $p=7$. Su complejidad computacional es O($n^2$) y es un algoritmo apropiado para problemas con muchas variables, muchas restricciones de desigualdad, grandes dominios y mas restricciones de inegualdad de las que maneja OFHH.

**Algoritmo heurístico POLYSA**

POLYSA [Salido'01c] es el algoritmo heurístico más complejo de todos. El poliedro inicial está formado por $n^2$ vértices aleatorios en cada faceta, por lo que en total consta de $2n^3$ vértices ($k=n^2$). Este algoritmo satisface el teorema de completitud y corrección para $k=n^2$ y $p=10$. Su complejidad computacional es O($n^3$) y es un algoritmo apropiado con muchas restricciones de desigualdad, dominios muy amplios, y varias variables y restricciones de inegualdad.

# 4 Evaluación del algoritmo MAS

En esta sección vamos a comparar el comportamiento de los algoritmos utilizados por MAS con algunas de las técnicas más actuales de resolutores de CSP's. Para ello hemos seleccionado Forward-checking (FC) y Real Full Look-ahead (RFLA)[1] por ser las técnicas que mejor comportamiento tienen en este tipo de problemas.

Para realizar las pruebas, se ha utilizado un PII-400 con 128 Mb. de RAM y sistema operativo Windows NT

Los problemas generados para hacer las pruebas dependían de cuatro parámetros significativos que condicionan el problema: $<v, c_{\leq}, c_{\neq}, d>$, donde $v$ es el número de variables del problema, $c_{\leq}$ es el número de restricciones de inegualdad, $c_{\neq}$ es el número de restricciones de desigualdad y $d$ es la longitud del dominio de las variables.

Los problemas fueron generados aleatoriamente modificando dichos parámetros siendo todas las restricciones del tipo expresado en (1), (2) y (3) con los coeficientes $p_i$ distintos de cero. Así cada una de las gráficas presentadas fija dos parámetros y varía el restante para poder estudiar el comportamiento de los algoritmos cuando se incrementa dicho parámetro.

Para realizar las pruebas se evaluaron 100 problemas para cada tipo de problema y valor de cada parámetro y presentamos el tiempo medio de CPU para cada una de las técnicas. A continuación presentamos cuatro grupos de gráficas (fig.2) correspondiente a los cuatro algoritmos que maneja MAS. Cada grupo resume el tiempo medio de CPU de cada técnica cuando se incrementan los parámetros significativos, que son: número de variables, número de restricciones de inegualdad y longitud del dominio. La gráfica que presenta el comportamiento de las técnicas cuando se incrementa en número de restricciones de desigualdad $c_{\leq}$ y $c_f$ no la mostramos por presentar un comportamiento análogo a las gráficas de longitud de dominios.

En general podemos observar que los algoritmos que MAS utiliza tienen un comportamiento mejor que las técnicas FC y RFLA. Estas dos técnicas tienen un comportamiento exponencial cuando se incrementa el número de variables, el número de restricciones y la longitud de los dominios. Sin embargo los algoritmos utilizados por MAS tienen un comportamiento lineal con el número de restricciones de inegualdad. Sin embargo estos

[1] Forward-checking y Real Full Look-ahead fueron obtenidos de CON'FLEX; un resolvedor en C++ que puede manejar problemas no binarios con variables acotadas en intervalos continuos. Este software puede encontrase en : http://www-bia.inra.fr/T/conflex/Logiciels/adressesConflex.html.

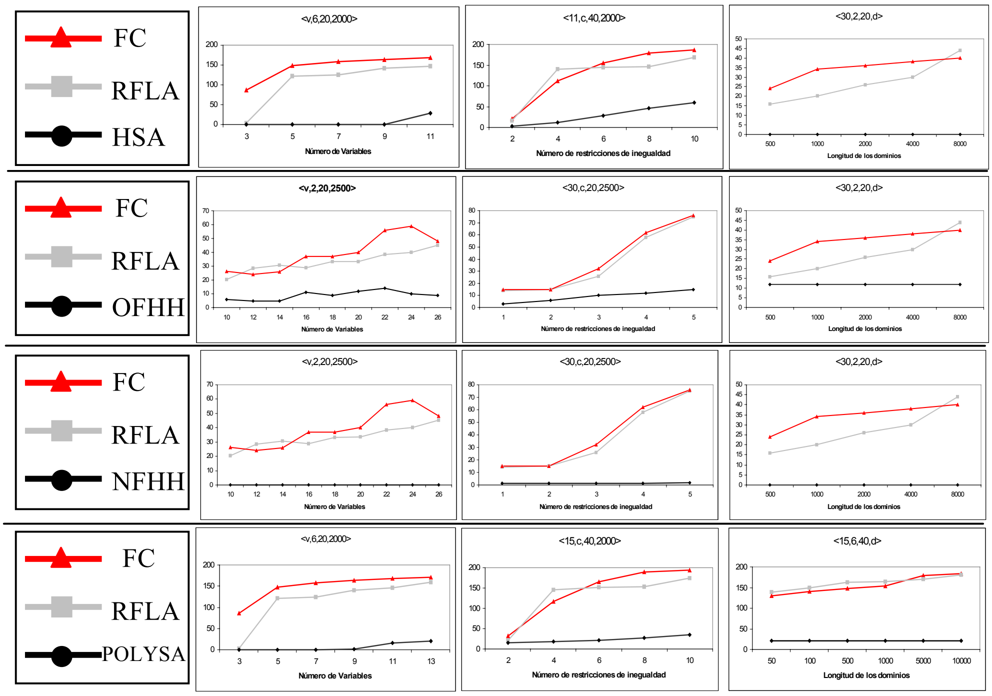


*Figura 2.* Tiempo Medio de CPU

algoritmos tienen un comportamiento constante con el número de restricciones de desigualdad y con la longitud de los dominios.

# 5 Conclusiones

En este artículo hemos propuesto un modelo llamado MAS como resolutor de TCSP's no binarios. Esta propuesta lleva a cabo el estudio de la consistencia mediante un poliedro que mantiene en sus vértices ($v$) asignaciones de valores a las variables que satisfacen todas las restricciones temporales del problema. Así las soluciones al TCSP son los vértices del poliedro así como cualquier combinación convexa entre dos vértices del poliedro que satisfagan todas las restricciones de desigualdad.

Este modelo es apropiado para problemas cuyas restricciones temporales son no binarias e incluso no lineales, y además se pueden insertar restricciones no binarias de forma incremental sin la necesidad de resolver todo el problema de nuevo.

Hay que tener en cuenta que este modelo maneja restricciones lineales, y trabaja con los vértices del poliedro resultante, al igual que otras técnicas como el método Simplex. Sin embargo el objetivo de un TCSP es totalmente distinto al que se obtiene con el método Simplex. Nuestro objetivo, además de manejar restricciones no lineales, es capaz de obtener información relevante como los dominios mínimos de cada variable, las solu

ciones que el usuario requiera, así como aquella solución que optimice una o varias funciones objetivo (multi-objetivo) que a la vez pueden ser lineales o no lineales.
En futuros trabajos trataremos de incrementar la expresividad de las restricciones para manejar problemas más complejos como problemas con restricciones disyuntivas.

# 6 Referencias

[Bacchus'98] Bacchus F., Van Beek P.: On the conversion between non-binary and binary constraint satisfaction problems. In Proc. AAAI-98, (1998) 311-318

[Bessière'99] Bessiere C.: Non-Binary Constraints. In Proc. (CP-99), (1999) 24-27

[Dechter'90] Dechter R.: On the expressiveness of networks with hidden variables. In Proc. AAAI'90, (1990) 556-562

[Larrosa'98] Larrosa J.: Algorithms and Heuristics for total and partial Constraint Satisfaction, Tesis Doctoral, Barcelona, Spain, 1998

[Rossi'90] Rossi F., Petrie C., Dhar V.: On the equivalence of constraint satisfaction problems. In Proc. ECAI-90, (1990) 550-556

[Salido'01] Salido M.A., Barber F.: An Incremental and Non-binary CSP Solver: The Hyperpolyhedron Search Algorithm. To appear in Proceedings of Seventh International Conference on Principles and Practice of Constraint Programming (Springer Verlag), (CP2001).

[Salido'01a] Salido M. A., Giret A., Barber F.: A Non-binary Constraint Satisfaction Solver: The One-face Hyperpolyhedron Heuristic. To appear in Proceedings of ES2001, the Twenty-first SGES International Conference on Knowledge Based Systems and Applied Artificial Intelligence. Published in Research and Development in Intelligent Systems XVIII. 2001.

[Salido'01b] Salido M. A., Giret A., Barber F.: Realizing a Global Hyperpolyhedron Constraint via LP Techniques. In Proceedings of Joint German/Austrian Conference on Artificial Intelligence (KI-2001) Workshop on New Results in Planning, Scheduling and Design (PUK2001). (2001) 78-88

[Salido'01c] Salido M.A., Barber F.: POLYSA: A Polinomial Algorithm for Non-binary Constraint Satisfaction Problems with <= and <>. To appear in EPIA'01 workshop on Constraint Satisfaction and Operational Research Techniques for Problem Solving (CSOR2001). (2001)