APLICACION DE CLP PARA MANTENER LA RAZÓN EN UN SISTEMA DE CONTROL CON RESTRICCIONES

Forradellas, R. *[A],[B]*
Ibañez, F. *[A],[B]*
Toledo, F. *[A],[B]*
Berlanga, R. *[A]*
Barber, F. *[A]*
Martin, G. *[A]*

[A] LISITT- Universidad de Valencia
C/Dr.Moliner 50 -46100 Burjassot -Valencia- España
FAX: +34-6-386 4568
E-mail: lisitt@evalun11.bitnet
lisitt@vm.ci.uv.es

[B] Universitat Jaume I
Campus de Penyeta Roja-Castellón-España
FAX: +34-964-345842
E-mail: forradel@inf.uji.es

RESUMEN:

En este trabajo se presentan diferentes métodos basados en técnicas de CSP [6] que permiten, mediante un lenguaje CLP, mantener la consistencia en un Sistema de Control con Restricciones sobre el que se realiza la aplicación.

Las diferentes técnicas están implementadas a través de aproximaciones realizadas en el Lenguaje CHIP [5], [7], [14] (Constraints Handling In Prolog) sobre las que se discuten las ventajas y desventajas que surgen entre ellas y con respecto a los métodos convencionales utilizados para el mantenimiento de la razón.

Palabras Clave: Mantenimiento de la Razón, Sistemas de Control, CLP.

Este trabajo fue parcialmente financiado por el Proyecto Esprit 5291 "CHIC" (Constraint Handling in Industry and Commerce), de la CEE.

1.- INTRODUCCION

Los Sistemas Basados en el Conocimiento y Razonamiento (SBCR) [16] tienen como objetivo *resolver problemas* por medio de la interacción de sus dos componentes principales: el *conocimiento*, visto como el conjunto de conceptos y sus interrelaciones, y el *razonamiento*, que es quien opera sobre el conocimiento, posibilitando que nuevos conceptos sean derivados a partir de otros.

El razonamiento se realiza cuando es posible inferir una información a partir de datos iniciales conocidos. Una de las formas más general de razonamiento es el No Monotónico. En este caso los Sistemas requieren un tratamiento especial, con el objeto de evitar los riesgos que pueden ocasionar una elección no correcta de soluciones. En otras palabras, el camino seleccionado en cualquier punto del árbol de búsqueda puede no conducir a la solución, en cuyo caso es necesario retornar a un punto anterior y reiniciar la búsqueda por otro camino. Este tratamieno, lo utilizaremos para encontrar soluciones mediante diferentes mecanismos de control propios del lenguaje CLP utilizados en nuestra aplicación [5], [7], [14].

Trataremos un Sistema de Control con Restricciones (SCR) en el cual es necesario mantener la razón, a partir de un conjunto inicial de creencias. Estas están almacenadas en la Base del Conocimiento, y están compuestas por *Suposiciones* y *Restricciones*. Para nuestro caso, el problema general consiste en mantener la razón del Sistema completo. Para lograr este objetivo se considera que el Sistema es Consistente, *si y sólo si* cada una de las Restricciones es consistente con el conjunto de Suposiciones. Cuando el Sistema se torna Inconsistente se debe producir una modificación de la información con el objeto de eliminar la contradicción. Esta modificación puede dar lugar a una inconsistencia motivada por la acción de otra restricción, por lo que nuevamente se producirán modificaciones y así sucesivamente.

El SCR que analizaremos es un Sistema de Control de Tráfico Urbano (SCTU) [1], [2], [3], [4], en el cual trataremos el problema de un *cruce urbano*, en principio de complejidad exponencial, que es el principal factor donde se produce explosión combinatoria. Esto implica trabajar con un árbol de búsqueda computacionalmente intratable con los métodos tradicionales y es donde hay que extremar esfuerzos para tornar tratable el problema y encontrar soluciones suficientemente cercanas al comportameinto real del sistema.

1.1.- Mantenimiento de la Razón

Convencionalmente, [11] los Sistemas para el Mantenimientode la Razón (SMR) son sistemas que permiten determinar un conjunto actual de creencias, a partir del conjunto actual de razones. El razonamiento se realiza a partir de las proposiciones almacenadas en la Base de Conocimiento -BC-. Cuando son encontradas nuevas razones, se actualiza el conjunto de creencias

El proceso que realiza el SMR consiste en explorar alternativas, realizar elecciones, explorar las consecuencias de las elecciones y por último comparar los resultados obtenidos cuando se utilizan elecciones diferentes. Si durante este proceso, se detecta alguna contradicción, el SMR actualiza la BC y cambia sus creencias con el propósito de eliminar la contradicción.

En la mayoría de los modelos desarrollados [11], [17], [18], [19], el SMR interactúa con un Solucionador de Problemas (SP), el cual transfiere al SMR las inferencias que realiza, y el SMR utiliza la estructura de estas inferencias para organizar las creencias del SP. La forma de interacción entre los dos subsistemas

varía, pero normalmente pueden considerarse que el SMR enviará información al SP sobre las creencias en que se basan las inconsistencias y sobre los conjuntos de creencias mutuamente consistentes con los que se puede trabajar. Toda la inferencia que se haga es comunicada al SMR por el SP. Por otro lado los modelos analizados no contemplan esta inferencia (cómo obtener nuevas creencias a partir de las existentes), siendo el SP quien contempla tal problema.

1.2.- Programación Lógica con Restricciones -CLP-

Muchos problemas de la vida real pueden ser tratatos el mediante manejo de restricciones. Tradicionalmente, la aproximación más común para resolver estos problemas consistía en tratar los problemas con lenguajes procedurales. Estas aproximaciones requieren sustanciales esfuerzos por parte del programador para el desarrollo de programas, y el producto terminado resulta difícil de mantener, modificar y/o adaptar. La programación lógica resulta muy apropiada en términos de declarativiidad, para el tratamiento de problemas con restricciones. Sin embargo, los lenguajes de programación lógica del tipo Prolog en la práctica resultan ineficientes para atacar problemas con explosión combinatoria considerable.

La Programación Lógica con Restricciones (Constraints Logic Programming, CLP) está basada sobre la idea de sustituir la unificación, la operación básica de los lenguajes lógicos tradicionales, por el concepto más general de resolución de restricciones sobre un dominio de computación. Una de las ventajas de los lenguajes CLP, es la facilidad de programación, lo que permite al programador razonar directamente en términos del dominio, en lugar de forzar la codificación a la semántica de objetos en términos del Universo de Herbrand. Otra ventaja es la eficiencia, que resulta del uso de algoritmos especializados para la resolución de restricciones sobre el nuevo dominio de computación. De aquí, resulta que CLP combina la declarativiidad y la flexibilidad de la programación lógica, con la eficiencia de aproximaciones procedurales.

Una razón para el éxito de CLP en recientes aplicaciones es la elección de dominios de computación integrados dentro de diferentes sistemas. En los últimos años, se han definido diversos lenguajes CLP. Estos, básicamente difieren en los dominios de computación sobre los que trabajan y/o los algoritmos para resolución de restricciones que usan.

1.2.1.- El Lenguaje CLP CHIP

CHIP (Constraints Handling In Prolog) es un nuevo lenguaje CLP que combina los aspectos declarativos de la programación lógica con la eficiencia de las técnicas de resolucion de restricciones [10], [12], [13]; fue desarrollado en el ECRC (European Computer-Industry Research Centre, fundado por BULL, SIEMENS e ICL) con el fin de abordar problemas del mundo real con un gran árbol de búsqueda y está basado en el uso activo de restricciones. De la versión original han surgido los productos: *Charme* de BULL, *Decision Power* de ICL, *Pecos* de ILOG y *CHIP* de COSYTEC, que es la versión que actualmente utilizamos en el marco del Proyecto Esprit "CHIC" [14].

De acuerdo con los criterios presentados, CHIP incluye tres nuevos dominios computacionales [8]: *booleanos*, *racionales*, y *dominios finitos*. Para cada uno de ellos se utilizan, diferentes técnicas de resolución de restricciones especializadas:

resolución de ecuaciones en el álgebra booleana, algoritmos tipo simplex, y técnicas de consistencia, respectivamente.

En nuestra aproximación utilizaremos *dominios finitos*, cuya característica básica es la de trabajar sobre *variables dominio*, que son variables con un rango de valores comprendido en un conjunto finito. CHIP diferencia entre dos clases distintas de tales variables, unas comprendidas en un rango de valores constantes y otras con un rango de valores comprendido en un conjunto finito de los números naturales.

Una declaración de dominios [5] puede verse en el siguiente predicado:

```
P(X,Y,Z):-X::1..9, Y::[1,2,7],...
```

En esta construcción, la declaración de dominios se emplea como un subobjetivo Prolog que tiene éxito y las variables libres que se colocan a la izquierda del operador "::" toman como dominio el expresado en la parte derecha de este operador. El predicado **P** tiene como primeros subobjetivos, las declaraciones de dominio de las variables **X** e **Y**. Estas variables pasan a ser inmediatamente variables dominio y podrán efectuarse sobre ellas todas las operaciones de reducción que se definan posteriormente. La variable **X** toma valores enteros sobre el intervalo [1,9], la variable **Y** toma valores en el conjunto {1,2,7}. Mientras que la variable **Z** (sobre la que no se ha definido ningún dominio) tomará sus valores en el universo de Herbrand.

Todas las restricciones implicadas en variables dominio son resueltas a través de técnicas de consistencia [6], [10], [12], [13]. La principal característica de estas técnicas es la de reducir el tamaño del espacio de búsqueda mediante los pasos siguientes:

Paso 1: Propagar las restricciones tanto como sea posible.
Paso 2: Realizar elecciones.

Estas técnicas de consistencia [13] ofrecen diferentes maneras de realizar la poda del espacio de soluciones, y se basan en las reglas de inferencia: Forward Checking (FCIR), Looking Ahead (LAIR) y Partial Looking Ahead (PLAIR).

2.- UN SMR CON RESTRICCIONES ABORDADO CON TECNICAS CLP

El mecanismo del SMR que vamos a proponer está enfocado a sistemas que almacenan la información mediante variables numéricas en el dominio de los naturales. La BC del SCR sobre el que es necesario mantener la razón, tiene almacenada un conjunto inicial de creencias (conocimiento inicial), compuesto por *Suposiciones* y *Restricciones*.

·*Suposiciones*: se definen como un tipo particular de Fórmulas Bien Formadas (FBF) o Proposiciones, las que se construyen por medio de un conjunto finito de disjunciones, de la forma:

$$A_j)\ \text{eq}\ (X_j,v_1)\ \text{or}\ \text{eq}\ (X_j,v_2)\ \text{or}\ \ldots.\ \text{or}\ \text{eq}\ (X_j,v_n).$$

Donde,

A_j es la suposición j de la base de suposiciones,
X_j es la j-ésima variable que describe el Sistema,
v_i $(0<i<n)$ son los valores constantes asociados a la variable X_j,
eq/2 es un predicado que liga la variable X_j con la constante v_i.

·*Restricciones*: se construyen a partir de operaciones relacionales entre dos tipos de términos (lineales y funcionales), que serán definidos a continuación.

Previo a describir la forma que tomarán las restricciones, haremos las siguientes definiciones preliminares:

- Sean $X_1..X_m$ variables lógicas.
- Para cada variable Xi definimos su clausura del siguiente modo:

$$cl(X_i) = \{a \,/\, eq(X_i,a) \models A_i\}$$

- Sean $F_1,..,F_k$ funciones definidas sobre variables lógicas de la forma

$$F_j: cl(X_j) \rightarrow n \quad (1 \leq j \leq k)$$

- Término Lineal -TL- se define recursivamente del siguiente modo:
  - Un número natural es un TL.
  - Si *c* es un número natural y X_i es una variable, entonces $c \cdot X_i$ es un TL.
  - Si T1 y T2 son TL entonces T1+T2 es un TL.
- Término Funcional -TF- se define recursivamente del siguiente modo:
  - Un número natural es un TF.
  - Si Xi es una variable, c es un número natural y Fj es una función como las que hemos definido al inicio de la sección, entonces c·F(Xj) es un TF.
  - Si T1 y T2 son dos TF, entonces T1+T2 es un TF.

La forma que tendrán las restricciones es:

Restricciones Tipo1

$$<TL_1> <op\_rel> <TL_2>$$

Donde: $<TL_1>$ y $<TL_2>$ son TL y <op_rel> es una de las relaciones: >=, >, <, <= o =

Veamos un ejemplo; sean las suposiciones:

A_1) $eq(X_1,1)$ or $eq(X_1,2)$

A_2) $eq(X_2,3)$ or $eq(X_2,4)$ or $eq(X_2,10)$

A_3) $eq(X_3,5)$ or $eq(X_3,6)$

A_4) $eq(X_4,7)$ or $eq(X_4,6)$ or $eq(X_4,7)$

Ejemplo: una Restricción de Tipo 1 sería: $2X_1 + 3X_4 = X_2 + 2X_3$

Restricciones Tipo 2

$$<TF_1> <op\_rel> <TF_2>$$

Donde: $<TF_1>$ y $<TF_2>$ son TF y <op_rel> ídem al anterior.

Por ejemplo: sean las suposiciones anteriores $A_1..A_4$, entonces una restricción posible sería: $2F_1(X_1) + 3F_2(X_2) = 5F_3(X_3) + 4F_4(X_4)$

Diremos que un Sistema de esta clase es consistente si y sólo si cada una de las restricciones, es consistente con el conjunto de proposiciones A_i. Para cada una de las restricciones descritas tendremos las siguientes situaciones:

Una restricción de Tipo 1 es consistente respecto al conjunto de Suposiciones si y sólo si Para todo i, $1 < i < k$ y para todo *a* , tal que $eq(X_i,a) \models A_i$ existen $v_1,...,v_{i-1},v_{i+1},...,v_m$ tal que,

$eq(X_1,v_1)$ and ...and $eq(X_{i-1},v_{i-1})$ and $eq(X_{i+1},v_{i+1})$ and ...and $eq(X_k,v_k)$
$\models A_1$ and A_2 and ... and A_{i-1} and A_{i+1} and ... and A_k.

y verifiquen la restricción dada, sustituyendo las variables

X_j, por los valores v_j $(1<j<m$ y $j=i)$
y X_i por *a*.

Para una restricción de Tipo 2 la consistencia respecto del conjunto de suposiciones se describe exactamente igual que para las restricciones de Tipo 1, excepto que se deben evaluar las funciones sobre las variables.

Cuando el SMR se torna inconsistente por algún motivo, debe producirse una modificación de la información de manera que hagamos desaparecer la contradicción aparecida. Esta modificación consiste en la eliminación de predicados eq/2 de las disjunciones que produzcan inconsistencia, es decir que no cumplan las condiciones de consistencia antes expresadas, a su vez esta modificación puede hacer inconsistente la hipótesis con alguna restricción produciendo una propagación de modificaciones sobre las hipótesis hasta que el sistema completo se torne consistente.

Si una restricción produce la eliminación de todos los predicados eq/2 de la misma suposición, entonces la restricción es incompatible con las suposiciones y deberá ser eliminada.

4.- APLICACIÓN

4.1.- El problema del Tráfico Urbano

El mecanismo actual para la gestión del tráfico en una red urbana [15] consiste en la asignación de tiempos de verde y rojo en cada una de las intersecciones por medio de *semáforos*. Un semáforo asigna de forma alternativa el derecho de paso a cada movimiento o grupo de movimientos que confluyen en una intersección. Esta asignación se realiza fijando dos valores: *Ciclo*, que es el número de segundos en que un semáforo en una intersección tarda en repetir su comportamiento y *Reparto*: que es el porcentaje de verde y rojo de cada uno de los movimientos posibles en una intersección para un determinado ciclo.

Mientras que el reparto está asociado a una intersección, el ciclo está asociado a un conjunto de intersecciones que se denominan Controladores de Ciclo Idéntico (CCI) que se encargan de regular el tráfico en una zona, de esta forma la red está dividida en zonas (CCI) y cada zona está compuesta por varias calles e intersecciones. De esta manera la gestión de tráfico en una ciudad puede ser reducida al manejo de un número de zonas de CCI. Para una eficiente gestión, el SCTU debe efectuar un análisis permanente del estado del tráfico, mediante detectores repartidos en la red, que permitirá la toma de decisiones en un sistema centralizado, él que transmitirá a los elementos reguladores de tráfico (semáforos), las soluciones encontradas.

El SCTU que tratamos incluye un simulador cuya función es la de proveer información de vital impotancia para la toma de decisiones. En este trabajo trataremos el problema que se presenta en una intersección con *m* entradas y *n* salidas, que es el principal factor donde se produce una explosión combinatoria sobre la cual realizaremos aproximaciones para el mantenimiento de la razón.

4.1.1.- Restricciones

En el Sistema presentado, el estado del mismo se describe mediante variables, que representan las densidades de vehículos, correspondientes a las entradas y salidas en una intersección, las cuales pueden tomar un número fijo de valores posibles. Estas variables (representadas en CHIP mediante variables dominio) están sometidas a verificar restricciones sobre densidades. Además a cada densidad le corresponde un flujo y por lo tanto, éstas variables tienen asociadas funciones con codominio real para representar el flujo. Mediante estas funciones se construyen TF, los cuales también deben verificar restricciones.

Cada vez que las variables dominio: $x_1, ..., x_{m+n}$ (para m entradas y n salidas) cambian sus valores, se deben satisfacer las siguientes restricciones:

- **De Transición** (RT): que dependen de los valores iniciales $x_1, ..., x_{m+n}$.
- **Instantáneas** (RI): que dependen sólo del valor de la variable_dominio asociada al al reparto (V), que toma valores en el conjunto F=$\{Fase_1, ..., Fase_k\}$ para k repartos.

Las RT restringen los posibles valores de las variables que representan las entradas y las salidas. Esta reducción se efectúa sólo una vez sin producir propagación [2], por lo tanto no es necesaria ninguna técnica de consistencia sofisticada para tratarlas. Las RI han sido formalizadas en anteriores presentaciones [2], [3], [4], [14] y se ha demostrado que para una determinada fase se representan mediante las fórmulas:

$$F(x_i) \doteq \sum_{j=1}^{m} \alpha_{ij} \cdot F(x_j) \qquad \text{para } m+1 \leq i \leq m+n$$

$$F(x_i) \doteq [0,0] \text{ para todas las entradas donde los vehículos no circulan.}$$

Estas fórmulas representan el flujo de las salidas en función del flujo de las entradas y serán representadas como Restricciones Tipo 2.

Donde: $F(x_i)$ es el intervalo $[F_{min}(x), F_{max}(x)]$, con F_{min} representando el mínimo flujo asociado a x, y F_{max} el máximo,
y la relación $\doteq$ entre 2 intervalos se define: $I_1 \doteq I_2$ Sii, $I_1 \cap I_2 = \varnothing$

4.2.- Aproximaciones del problema con CHIP

Las hipótesis pueden ser representadas en CHIP mediante *variables dominio*. Definiremos una variable por cada hipótesis, siendo su dominio, los posibles valores que tenía asociada la variable a partir de los predicados eq/2 que constituían dicha hipótesis, definido anteriormente como clausura de una variable. Para el tratamiento de las restricciones de Tipo 2 se utilizarán las reglas de inferencia FCIR y PLAIR cuyos mecanismos de control se describen brevemente:

4.2.1.- Forward Checking -FC-

Este es un procedimiento de búsqueda eficiente, especializado para Constraints Satisfaction Problems (CSPs) [9] que consiste en reducir el conjunto finito de valores posibles que puedan ser asignados a una variable. De esta manera se poda el espacio

de búsqueda *a priori* para eliminar la combinación de valores que no pueden dar lugar a una solución del problema.

Esta regla de inferencia permite el mecanismo de propagación de las restricciones, las que tienen la propiedad de que permanecen en espera hasta tener una sola variable_dominio no instanciada, estando el resto de las variables que forman parte de la restricción, instanciadas. Con ésta metodología podemos resolver eficientemente el problema planteado y veremos como se producen las reducciones sobre las variables dominio asociadas a las RI. Estas se basan en las condiciones que deben satisfacerse para que la intersección de dos intervalos sea no nula [4]. Estos intervalos están configurados de la siguiente manera: el primero corresponde al intervalo de flujo asociado a la densidad de una salida, y el segundo corresponde a la suma de los intervalos asociados a las entradas, los cuales están multiplicados cada uno ellos, por el coeficiente que asocia cada entrada con cada salida. Estos coeficientes forman una matriz que será distinta para cada fase

Dados los intervalos I_1 e I_2, $I_1 \cap I_2 = \varnothing$ Sii

[A] $F^-(I_2) > F^+(I_1)$ o [B] $F^+(I_2) < F^-(I_1)$

donde $F^-(I)$ (respectivamente $F^+(I)$) representa el extremo inferior (superior) del intervalo I. Así si [A] se cumple podemos asegurar que la intersección es vacía. El intervalo asociado a las entradas, es la suma de los intervalos asociados a cada entrada multiplicado por los correspondientes coeficientes, por lo tanto, suponiendo que I_1 representa una salida, si I_2 representa una sola entrada y la condición [A] se cumple, podemos asegurar que seguirá siendo cierta si I_2 representa todas las entradas, siendo posible, de esta forma, "podar" el espacio de búsqueda "a priori". Es por ello que podemos aplicar a cada pareja entrada-salida la condición [A] y descartar las densidades que lo cumplan, ya que es seguro que no producirán intersección. De esta manera podemos generar un conjunto de restricciones binarias que asociarán cada salida con cada entrada

4.2.2.- Partial Looking Ahead -PLA-

Looking Ahead (LA) es otro procedimiento de búsqueda utilizado en los problemas de combinatoria cuyo método difiere en el modo en que las restricciones pueden ser usadas; para el caso del FC sólo una variable debe estar desinstanciada, mientras que en LA pueden haber varias variables desinstanciadas. Para este caso el posible conjunto de variables es reducido, pero las restricciones no son totalmente satisfechas y pueden ser reconsideradas más tarde [12].

En otras palabras, el camino seleccionado en cualquier punto del árbol de búsqueda puede no conducir a la solución, en cuyo caso es posible retornar al punto original y encaminar la búsqueda por otro camino. Esta modificación puede dar lugar a una Inconsistencia motivada por la acción de otra restricción, por lo que nuevamente se producirán modificaciones y así sucesivamente hasta que el Sistema completo se torne Consistente.

El mecanismo de control PLA provee una base formal para la implentación de algunas restricciones. Esta implementación no es un caso particular de FC ni de LA, pero es posible considerarla como un caso intermedio entre ellos. La regla de inferencia PLAIR [5], [6] consiste en reemplazar algunos puntos, en la definición de LAIR utilizada en LA, para lo cual el conjunto de valores no son definidos en la regla

de inferencia y son dependientes de cada restricción en particular. Para este caso las restricciones tienen la propiedad de reducir el dominio de las variables, para que satisfagan la restricción. En esta aproximación utilizaremos el predicado "*element*" preuefinido en CHIP, para resolver el problema [5], [14].

- Supongamos una posible ecuación para las RI: $F(x_3) \doteq 0 \cdot F(x_1) + 0.5 \cdot F(x_2)$
  y las variables dominio: $\_OFmin_3$, $\_OFmax_3$, $\_IFmin_1$, $\_IFmax_1$, $\_IFmin_2$, $\_IFmax_2$
  para representar, $Fmin(x_3)$, $Fmax(x_3)$, $Fmin(x_1)$, $Fmax(x_1)$, $Fmin(x_2)$, $Fmax(x_2)$
- Obtenemos: $[\_OFmin_1, \_OFmax_1] = [0*\_IFmin_1+0.5*\_IFmin_2, 0*\_IFmax_1+0.5*\_IFmax_2]$

Considerando a *F(x)* y la relación entre 2 intervalos definidos en 4.1.1 queda:

$\_OFmax_1 \geq 0*\_IFmin_1 + 0.5*\_IFmin_2$ and $\_OFmin_1 \leq 0*\_IFmax_1 + 0.5*\_IFmax_2$ [*]

La solución se alcanza usando un par de predicados *element* para cada variable, uno para establecer la correspondencia con el flujo mínimo y otro para el flujo máximo. Por ejemplo, para x_1:

a) element(x_1,[0,90,265,...],$\_IFmin_1$) y **b)** element(x_1,[90,265,375,...],$\_IFmax_1$)

donde la lista [0,90,265,...], representa [$Fmin(x_1)$, $Fmin(x_2)$, $Fmin(x_3)$, ...],
y la lista [90,265,375,...], representa [$Fmax(x_1)$, $Fmax(x_2)$, $Fmax(x_3)$, ...]

multiplicados por el valor 100, puesto que CHIP trabaja sólo con naturales, las restricciones [*] se reescriben en CHIP de la siguiente forma:

100 * _OFmax1 #>= 0 * _IFmin1 + 50 * _IFmin2

Ahora, las reducciones se hacen sobre variables dominio que representan flujos. Si $\_IFmin_1$ es reducida, entonces, x_1 será reducida como concecuencia del predicado *element* **a)**, pero esta redución despertará el predicado *element* **b)**, y consecuentemente será reducida la variable $\_IFmax_1$. Ahora se despertarán todas las restricciones donde aparece $\_IFmax_1$. y así sucesivamente, hasta tornar el sistema Consistente.

5.- CONCLUSIONES

En la aproximación implementada con FC, el tamaño del árbol de búsqueda es reducido, aunque provoca un gasto de tiempo adicional en cada nodo, la cantidad de nodos considerados se reduce y de aquí el número de restricciones manipuladas. De este manera, mediante la poda "a priori" antes del descubrimiento del fallo, se logra conseguir una notable ventaja frente al método de Bactraking Standard, Pero hay partes del árbol que no verifican las RI y sin embargo no son podadas, puesto que sólo estamos usando una condición suficiente.

En cuanto a la aproximación implementada con PLA el árbol es completamente "podado", puesto que usamos una condición necesaria y suficiente. De esta manera todas las soluciones generadas verifican las RI y viceversa. Con el uso de esta aproximación se logra además una forma más natural de resolver el problema.

Se logra un considerable ahorro de tiempo, pues si consideramos que el árbol inicial de búsqueda tiene 8^{n+m} nodos (n entradas y m salidas), al efectuar una "poda" inicial sobre las variables dominio, mediante las Restricciones del Sistema, se reducen los dominios de dichas variables y en consecuencia el número de combinaciones

posibles. El árbol inicial queda sustancialmente reducido. Por último, aunque el coste adicional en cada nodo del árbol es mayor, para un gran árbol de búsqueda la eficiencia de las restricciones supera ampliamente el coste adicional mencionado.

En el SMR presentado se obtienen resultados que presentan ventajas frente a la forma de abordar el problema utilizando metodologías tradicionales. Por un lado la metodología presentada, permite representar el problema en forma sistemática y declarativa con CLP. Por otro lado, se logra un considerable ahorro de tiempo de ejecución debido a dos razones: 1ra. PLAIR permite trabajar en un espacio de búsqueda sustancialmente reducido y 2da. el trabajo realizado en cada nodo, es obtenido con mínimo coste computacional debido al mecanismo asociado a PLAIR.

6.- AGRADECIMIENTOS

Parte de la investigación realizada en el presente trabajo ha sido realizada mientras los autores trabajaban en el Proyecto Esprit 5291 de la CEE "CHIC" [14] Organizaciones colaboradoras son: ECRC y SIEMENS (Alemania), ICL e Imperial College (Gran Bretaña), BULL, Reanault, CERT/ONERA y Dassault (Francia), AIS y Braghenti (Italia), CMSU (Grecia) y OCT (España).

7.- REFERENCIAS

[1] Martín G., Toledo F. & Moreno S. "KBS for Road Network Traffic Control State of the Art and Perspectives", AI & ES, Avignon '92.
[2] Ibañez F., Forradellas R., Toledo F., Martín G. y Moreno S. "Aplicación de CLP a la Simulación Cualitativa de la Gestión de Tráfico Urbano", AEPIA '91.
[3] Martín G., Toledo F., Ibañez F., y Forradellas R. "User Requirements Report for Urban Traffic Control in CHIP", CHIC Project, Technical Report, Munich 11/91.
[4] Ibañez F. Forradellas R. Berlanga, R. Toledo F. y Martín G. "Propagación de Constraints sobre el Tratamiento de un Cruce Urbano" Jornadas ProDe91.
[5] CHIP Reference Manual, ICL/COSYTEC 1991/92
[6] Van Hentenryck, P "Constraints Satisfaction in Logic Programming", MIT 1989.
[7] Forradellas R., Ibañez F., Berlanga R., Martín G. y Toledo F. "Manejo de Constraints en CHIP", Jornadas ProDe91.
[8] Van Hentenryck P., Dincbas M., "Domains in Logic Programming", IAAA 86
[9] Van Hentenryck P., Dincbas M., "Forward Checking in Logic Programming", FICLP, 5/87
[10] Dincbas M., Van Hentenryck P., Simonis H., Aggoun A., Graf T. and Berthier F "The Constraints Satisfaction Logic Programming Languaje CHIP", FGCS 88
[11] Botti V."Mantenimiento de la Razón en Sistemas Basados en el Conocimiento Dependientes del Tiempo", Tesis Doctoral, UPV, 1990
[12] Van Hentenryck P. "A Theorical Framework for Consistency Techniques in Logic Programming", IJCAI 87.
[13] Dincbas M., Van Hentenryck P., Simonis, "Extending Equation Solving and Constraint Handling In logic Programming", CREAS 87.
[14] Esprit "CHIC" Project, Second Review, Paris 1/92
[15] "Traffic Engineering Handbook", Prentice-Hall, 1989
[16] Rich, Elaine, "Artificial Intelligence", McGraw-Hill 1989
[17] Doyle J. "A Truth Maintenance System" Artificial Intelligence, 1979.
[18] De Kleer J., "An Assumtion-based TMS", Artificial Intelligence, 1986.
[19] McDermontt D., "Data Dependencies on Inequalities", AAAI-83.