

# Proyecto de Investigación Aplicada:CSP

## Inteligencia Artificial

### Primer Semestre, Año 2004

Daniel Basterrica Brockman  
<aruns@labsc.inf.utfsm.cl>

15 de abril de 2004

#### Resumen

El presente documento explica el problema CSP (Constraint Satisfaction Problem), e incluye las líneas guías para el trabajo de investigación que deberán realizar los alumnos que les corresponda trabajar con este problema. Las bases incluidas en este documento son las directivas por las cuales será evaluado el trabajo realizado, incluyendo la descripción del problema y la forma específica en que se trabajará en esta ocasión, las explicaciones sobre el informe a entregar, y algunas referencias útiles para conocer mejor el problema.

## 1. Introducción

Constraint Satisfaction Problem generaliza el modelo de los problemas de satisfacción de Restricciones. Su aplicación por tanto es muy amplia: Coloreo de Grafos, la versión CSP del problema del Vendedor Viajero, el problema de las 8 Reinas, Space Planning con espacio fijo, reconocimiento de patrones, problemas de Robotica como el desplazamiento sin colisionar, y muchos otros.

Tratar generalizadamente el problema permite trabajar con casos particulares que simplemente deben ser descritos de manera genérica, es decir, utilizar la notación general de CSP, donde cada restricción es escrita por extensión.

Solucionar un CSP significa encontrar una combinación de asignaciones, tal que ninguna restricción sea violada o insatisfecha. Para trabajar este problema como un problema de optimización, se fija una función de evaluación relacionada con el costo de no satisfacer todas las restricciones, es decir, para este trabajo se considerará el costo como una función directamente proporcional a la cantidad de restricciones sin satisfacer por la solución encontrada. Se puede decir entonces que un problema CSP será resuelto cuando el costo de CSOP (Constraint Satisfaction Optimization Problem) que representa el mismo problema es igual a cero.

Se debe recordar que un problema en particular podría no tener solución, y eso en general, es más difícil de demostrar que encontrar una solución.

## 2. Definición del problema

El problema genérico CSP, se define como una tupla  $(X, D, C)$ , donde  $X$  es un conjunto finito de variables,  $D$  es una función que mapea cada variable  $X_i \in X$  al dominio  $D(X_i)$ , que es un conjunto finito de los posibles valores que se le puede asignar a  $X_i$ , y  $C$  es un conjunto de restricciones que describen por relaciones de algunas variables, la lista de los valores que no pueden ser asignados simultáneamente a cada una de ellas. En este caso, se trabajará con **CSP's binarios**, lo que significa que las relaciones serán únicamente entre dos variables.

### 2.1. Solución

Una asignación es un conjunto de pares que representa la asignación simultánea de los valores  $v_i$  a cada variable  $X_i$ , de la forma  $A = (\langle X_1, v_1 \rangle, \dots, \langle X_n, v_n \rangle)$ , para las  $n$  variables del problema.

### 2.2. Insatisfacción de restricciones

Las violaciones o insatisfacción de restricciones ocurren cuando una asignación utiliza alguna de las combinaciones de asignaciones descritas en el conjunto  $C$ . Esto implica un  $costo(A)$ , que será igual a la cantidad de restricciones insatisfechas.

### 2.3. Parámetros del problema

Los problemas CSP binarios pueden ser generados aleatoriamente. Como un problema CSP puede ser de distintos tamaños y niveles de dificultad, se definen parámetros que permiten clasificarlos (y generarlos):

$n$ : Número de variables del problema.

$m$ : El tamaño del dominio.

$p1 \in (0, 1)$ : Es la medida de conectividad (definiendo la cantidad de restricciones).

$p2 \in (0, 1)$ : Es la medida de incompatibilidad (definiendo el número de valores incompatibles por cada restricción).

Estos parámetros permiten utilizar generadores para crear problemas aleatorios, pero con una cantidad configurable de variables, tamaño de dominio, conexiones y grado de incompatibilidad. Estos dos últimos parámetros tienen dos formas de ser vistos, determinística o probabilísticamente, es decir,  $p1$  puede determinar la cantidad exacta de restricciones, en porcentaje del máximo de conectividad (que todas las variables estén conectadas con todas, i.e.,  $n(n - 1)/2$  restricciones); o la probabilidad de que dos variables estén conectadas por una restricción. Así mismo,  $p2$  puede indicar la cantidad exacta de valores incompatibles en cada restricción, en porcentaje del máximo de combinaciones  $m^2$ ); o la probabilidad de que dos valores sean incompatibles.

Para este trabajo, se utilizarán ambos parámetros probabilísticamente.

### 3. Entradas y salidas

La definición de un problema en particular, y que será el formato de entrada para el algoritmo de solución es el siguiente:

Por cada restricción, se listan las dos variables involucradas, y todos los pares de valores incompatibles, de la siguiente forma:

$$X_i \quad X_j \quad v_{ia} \quad v_{jb} \quad v_{ic} \quad v_{jd} \quad -1$$

Donde  $X_i$  y  $X_j$  son las dos variables conectadas, y tienen los pares de valores incompatibles  $\langle v_{ia}, v_{jb} \rangle, \langle v_{ic}, v_{jd} \rangle$ . Además, al terminar de listar los valores incompatibles se tiene un  $-1$ . Finalmente, otro  $-1$  finaliza el ingreso de restricciones. El orden de las restricciones es creciente respecto al nombre de la variable  $X_i$ , y luego la variable  $X_j$ , siendo además siempre  $X_j > X_i$  (matriz diagonal).

Por ejemplo:

$$\begin{array}{cccccccc} 3 & 4 & 1 & 1 & 1 & 2 & 1 & 3 & -1 \\ 4 & 5 & 0 & 0 & 2 & 4 & 3 & 4 & 3 & 2 & -1 \\ -1 \end{array}$$

En este ejemplo, las restricciones son las dos siguientes:

- Las variables 3 y 4 no pueden tener los valores 1 y 1 respectivamente, 1 y 2, ni 1 y 3.
- Las variables 4 y 5 no pueden tener los valores:  $\langle 0, 0 \rangle, \langle 2, 4 \rangle, \langle 3, 4 \rangle$  ni  $\langle 3, 2 \rangle$ .

Para el trabajo a realizar, se entregarán problemas definidos y se dispondrá de generadores de problemas. Los programas que implementen los algoritmos de solución deberán leer el problema desde archivos que poseerán la estructura descrita, y entregar la mejor solución encontrada y su costo.

## 4. Resultados y gráficos

Los resultados básicamente deberán estar expuestos en gráficos, si es prudente, incluir tablas resúmenes de datos (pequeñas) con el objetivo de poder esclarecer los valores exactos, o completar ideas.

Para las pruebas y experimentos se debe tomar en cuenta que no todos los computadores tienen la misma velocidad, por lo tanto, el tiempo que demoró no es tan buen indicador como la cantidad de iteraciones realizadas. Se debe describir el hardware y software usado, incluyendo los generadores aleatorios que hayan sido seleccionados para trabajar los aspectos randómicos.

Los análisis y las conclusiones son tanto o más importantes que los datos en sí, por lo tanto ambos deben ser incluidos de manera clara y concisa.

Algunos gráficos sugeridos:

- Convergencia del costo respecto a la cantidad de iteraciones en la solución de algún problema (*Costo vs. Iteraciones*).
- Costo o distancia al óptimo respecto a instancias de problemas con diferente tamaño o dificultad (*Costo vs. dificultad/Distancia al óptimo vs. dificultad*).
- Cantidad o porcentaje de problemas resueltos ( $costo = 0$ ) respecto al tipo de problema (tamaño/dificultad) (*Porcentaje de problemas resueltos vs. tipo de problema*).
- Comportamiento respecto a la variación de los parámetros.

## 5. Referencias útiles

- <http://www.lirmm.fr/~bessiere/generator.html>
- [cyl.cs.cf.ac.uk/research/pub/constraints.pdf](http://cyl.cs.cf.ac.uk/research/pub/constraints.pdf)
- [http://homepages.cwi.nl/~jvhemert/publications/cec2003.Evolving\\_Binary\\_Constraint\\_Satisfaction\\_Problem\\_Instances\\_that\\_are\\_Difficult\\_to\\_Solve.pdf](http://homepages.cwi.nl/~jvhemert/publications/cec2003.Evolving_Binary_Constraint_Satisfaction_Problem_Instances_that_are_Difficult_to_Solve.pdf)