Satisfacción de restricciones: resolver crucigramas desde otra perspectiva

Víctor Noriega, Fabián Encinas, Mario Castro

***Abstract*— We had resolved a lot of crosswords in our lifes. But, what about the modeling of crosswords? Through this document we will try to show you how to create any crossword with a determined number of spaces and words through constraint satisfaction process from the perspective of just computer science students.**

I. INTRODUCCIÓN

La intención de este mini-reporte es conocer, de los crucigramas, cinco cosas principalmente:

1) La representación del conjunto de variables.
2) El dominio de cada variable.
3) Los vecinos de cada estado.
4) Las restricciones binarias.
5) ¿Hay restricciones globales? Si las hay, ¿cuáles son?

Trataremos en la medida de lo posible explicar los razonamientos detrás de las decisiones tomadas (porqué una representación y no otra) y en algunos casos agregar pseudo-código muy parecido a Python (porque un código elegante vale más que mil imágenes).

II. CONJUNTO DE VARIABLES: CONJUNTO DE TUPLAS DE TUPLAS

El llamado conjunto de variables en los CSP, es uno de los principales tres componentes de tales problemas, que suele definirse como $\{X_1, X_2, ..., X_n\}$ y cuando asignemos valores a algunas o a todas las variables, entonces decimos que tenemos un estado. Particularmente para el caso del crucigrama, hemos pensado en la siguiente representación ”formal”:

$$X = \{(V_1, V_2, ..., V_{n_{max}}), (H_1, H_2, ..., H_{m_{max}})\}$$

donde $V_i = (p_x, p_y)$ que a su vez, representan la posición en x en el tablero, la posición y en el tablero (obviamos que para H_i es lo análogo). Los valores que pueden tomar cada una de estas variables es un conjunto de palabras, que a su vez está partido en dos subconjuntos: las verticales y las horizontales.

III. DOMINIO DE LAS VARIABLES

A cada variable se le asigna un valor, y con una variable que tenga asignado un valor tenemos un estado. Pero, ¿qué valores puede tomar una variable? A este conjunto le conocemos cono el dominio de las variables y vamos a definirlo como sigue para las variables verticales:

$$D_i = \{([0 - (n_{max} - len(x_i)), ([0 - m_{max}])\}$$

donde $len(x)$ es una función que va del conjunto de la parte de cadena de cada variable hacia los enteros, y representa el largo de la cadena de caracteres.

También nótese que hacemos uso de la notación de rango en los conjuntos para especificar que las posiciones pueden ir desde 0 hasta algún $k \in \mathbb{N} \cup 0$ (también obviaremos el dominio para las horizontales, pues vuelve a ser lo análogo).

Básicamente estamos diciendo que el dominio de las variables, es una dupla representando la posición, y los valores que estas van a tomar son cadenas de caracteres.

IV. VECINOS DE X_i

¿Qué es un vecino de una variable? Nosotros definimos un vecino como un estado con una nueva asignación. Por supuesto, el estado con menos asignaciones es a su vez vecino de aquel que tiene más asignaciones. De manera que las palabras verticales pueden ser vecinas de otras verticales. Claro, estamos considerando incluso un estado con sólo palabras verticales u horizontales, pero eso es interés de la solución, no de las vecindades de un estado, y para llegar a esa solución hay que hablar de las restricciones.

V. RESTRICCIONES BINARIAS

¿Restricciones en el crucigrama? Varias. Toda restricción es posible transformarla a una binaria, es decir, la relación que existe entre una variable y otra. Es por esta razón que en muchas ocasiones (por lo menos eso dice el texto de Norvig y Rusell) se suelen dejar sólamente restricciones binarias, tal como es nuestro caso.

Sean x_i, v_i, x_j, v_j donde las x son las variables de nuestro conjunto, y las v son los valores de tales variables, entonces podemos definir las restricciones binarias a partir del siguiente pseudo-código (ver sig. página)

VI. RESTRICCIONES GLOBALES: ¿LAS HAY?

Vamos a entender por restricción global cierta restricción que tiene que involucrar necesariamente un número arbitrario de variables. En nuestro caso particular, en un crucigrama debemos hacer que todas las palabras se crucen unas con otras. Yo considero muy viable representar el crucigrama en un grafo no dirigido, donde los nodos sean las palabras que ya han sido asignadas y las aristas sean la relación de cruza entre una palabra y otra. De manera que, si es posible llegar de un nodo a cualquier otro, se cumple la restricción global. Para verificarlo, se puede hacer también una busqueda con back-tracking.

```
def restric_binarias(crucigrama, xi,vi,xj,vj):
    """
    Vamos a restringir a que las variables
    no se puedan solapar, que haya
    al menos dos de separacion entre ellas,
    y que haya almenos una coincidencia
    @param crucigrama: un objeto tipo crucigrama
    @param xi: una variable
    @param vi: posible valor para la variable xi
    @param xj: otra variable
    @param vj: posible valor para la variable vj
    @return: un booleano que indica si se cumplen
    todas las restricciones binarias
    """

    if vi is in crucigrama.vert and vj is in
    crucigrama.vert:
        return (abs(xi[0]-xj[0] == 1) and (xi[1]>xj[1] and
                xj[1]+len(vj)>= xi[1]-1 or xj[1]>xi[1]
                and xi[1]+len(vi)>= xj[1]-1))

    elif vi is in crucigrama.hor and vj is in
    crucigrama.hor:
        return (abs(xi[1]-xj[1]==1) and (xi[0]>xj[0] and
                 xj[0]+len(vj)>= xi[0]-1 or xj[0]>xi[0]
                 and xi[0]+len(vi)>= xj[0]-1))
    else:
        return vi.coincidencia(vj)
        #Coincidencia es un metodo que verifica si hay
        #letras que coinciden entre dos cadenas
```