

# Técnicas, Entornos y Aplicaciones de Inteligencia Artificial

## Práctica 3. Problemas de Satisfacción de Restricciones

**Objetivo:** Modelar y resolver problemas CSP, utilizando el entorno MiniZinc



### MiniZinc:

- Entorno de desarrollo para la edición de modelos basados en. Restricciones.
- Compilación del modelo en FlatZinc > Diversos resolvers (GECODE)
- Disponible (libre y código abierto) : <https://www.minizinc.org/>  
*Windows / MacOS / Linux* <https://github.com/MiniZinc/MiniZincIDE/releases/>
- Ampla documentación: (Tutorial, Manual del Usuario, Manual de Referencia) ⇒ Handbook

# Interfaz MiniZinc

The screenshot displays the MiniZinc IDE interface. At the top is a menu bar with File, Edit, MiniZinc, View, and Help. Below it is a toolbar with icons for New model, Open, Save, Copy, Cut, Paste, Undo, Redo, Shift left, Shift right, Run, Solver configuration (set to Gecode 6.1.0 [built-in]), Show configuration editor, and Show project explorer. The main editor window shows a MiniZinc model named 'send-more-money.mzn' with the following code:

```
1 include "alldifferent.mzn";
2 var 1..9: S;
3 var 0..9: E;
4 var 0..9: N;
5 var 0..9: D;
6 var 1..9: M;
7 var 0..9: O;
8 var 0..9: R;
9 var 0..9: Y;
10 constraint 1000 * S + 100 * E + 10 * N + D
11 + 1000 * M + 100 * O + 10 * R + E
12 = 10000 * M + 1000 * O + 100 * N + 10 * E + Y;
13 constraint alldifferent([S,E,N,D,M,O,R,Y]);
14 solve satisfy;
15 output [ " ", show(S), show(E), show(N), show(D), "\n",
16 "+ ", show(M), show(O), show(R), show(E), "\n",
17 "= ", show(M), show(O), show(N), show(E), show(Y), "\n"];
```

Below the editor is an 'Output' window showing the compilation and execution results:

```
Compiling send-more-money.mzn
Running send-more-money.mzn
9567
+ 1085
= 10652
-----
Finished in 256msec
```

At the bottom left, the status bar indicates 'Line: 2, Col: 1'.

Annotations on the image:

- Ejecución** (Execution) points to the Run button in the toolbar.
- Configuración Resolvedor** (Solver Configuration) points to the Solver configuration dropdown and the Show configuration editor button.
- Edición del Modelo** (Model Editing) points to the main code editor area.
- Salida Ejecución** (Execution Output) points to the Output window.

## Notas:

- Todas las líneas acaban con ;
- %: Símbolo de comentario (hasta final línea)
- Dependiente de mayúsculas/minúsculas (Identificadores y nombres de variables)
- Identificadores de variables empiezan por letra y pueden contener números, letras y \_.
- Debe evitarse en lo posible en uso de variables reales (dominios continuos)
- La configuración de las ventanas es modificable.

## Configuración Resolvedor (GECODE)

- (1) Resolvedor a utilizar : Gecode.
- (2) Tiempo máx. de resolución (0)
- (3) Nº Soluciones. Por defecto:  
en optimización, soluciones intermedias,  
en satisfabilidad, la 1ª solución
- (4) Nivel de preproceso:  
básico (O1 – O3)  
nodo consistencia (O4)  
arco consistencia (O5)
- (5) Opciones a de salida.  
Se recomienda 'Clear output before each run'.

The screenshot shows the 'Configuration' dialog for Gecode. It is divided into several sections:

- Gecode 6.1.0 [built-in]**: Contains 'Clone', 'Reset to defaults', and a checked 'default' checkbox. A green box with the number '1' is next to the 'default' checkbox.
- Solving**:
  - Solver**: Set to 'Gecode 6.1.0'.
  - Time limit**: Set to '0 seconds (disabled)'. A green box with the number '2' is next to the '0'.
  - Default behavior**: Includes options for 'Optimization problems' (print all intermediate solutions) and 'Satisfaction problems' (stop after first solution).
  - User-defined behavior**: A green box with the number '3' is next to the section header. It includes a checkbox for 'Print intermediate solutions (for optimization problems)' and a field for 'Stop after this many solutions (0 means "all")' set to '1'.
- Compiler options**: A green box with the number '4' is next to the section header. It includes checkboxes for 'Verbose compilation' and 'Output statistics for compilation', a dropdown for 'Compiler optimisation level' set to '-O1 (default)', and fields for 'Additional data' and 'Additional compiler arguments'.
- Solver options**: A green box with the number '5' is next to the section header. It includes a field for 'Number of threads' set to '1', a checkbox for 'Random seed', a field for 'Additional solver command line arguments', and checkboxes for 'Free search', 'Verbose solving', and 'Output statistics for solving'.

## Especificación Modelo CSP

### % Esquema de un Modelo CSP en MiniZinc

```
include "alldifferent.mzn";      % Inclusión código restricciones especiales
include "datos1.dzn";           % Inclusión datos
```

```
int a;                          % Parámetros. Valor por asignación, fichero externo o interfaz.
var int: b;                     % Variables tipadas float|int|bool|string/enum
var 0..100: c;
```

```
constraint 250*b + 200*c <= 10*a; % Restricciones (aritmético-lógicas)
constraint .....
```

```
solve maximize 400*b + 450*c;    % Objetivo resolutor (solve satisfy; por defecto)
```

*% Formato de salida (asignaciones a las variables)*

```
Output ["Resultado b= ", show(b), "\n",
        "Resultado c = ", show(c)];
```

## Especificación Modelo CSP

% Modelo ejemplo (Nº de pasteles de plátano y chocolate)

**int** b;      % **Parámetro**. Numero de pasteles de plátano  
**var** 0..100: c;      % **Variable**. Numero de pasteles de chocolate

% gramos de harina  
**constraint** 250\*b + 200\*c <= 4000;  
% numero de platanos  
**constraint** 2\*b <= 6;  
% gramos de azucar  
**constraint** 75\*b + 150\*c <= 2000;  
% gramos de mantequilla  
**constraint** 100\*b + 150\*c <= 500;  
% gramos de cacao  
**constraint** 75\*c <= 500;

% maximizar cantidad ponderada pasteles  
**solve maximize** 400\*b + 450\*c;

**output**

[ "no. of bananas cakes = ", **show**(b), "\n",  
  "no. of chocolate cakes = ", **show**(c), "\n"];

**Tipos:** float/int/bool/string/enum

### **Restricciones (aritmético-lógicas)**

Operad. relacionales: = (==), !=, >, <, <=, >=  
Operad. aritméticos: +, -, \*, /, div, mod, pow  
Func. aritméticas: abs, sqrt, pow, ...

**solve** satisfy; %por defecto  
**solve** maximize {arithmetic expression};  
**solve** minimize {arithmetic expression};

**output** [ {string|expr}, . . . . . ];  
expression: **show** (var)  
"\\n": Nueva línea  
"\\t": Tabulación

## Parámetros y Fichero de datos

% Modelo con ficheros de datos

```
include "datos1.dzn";
```

%Parametros con valor adquirido por fichero

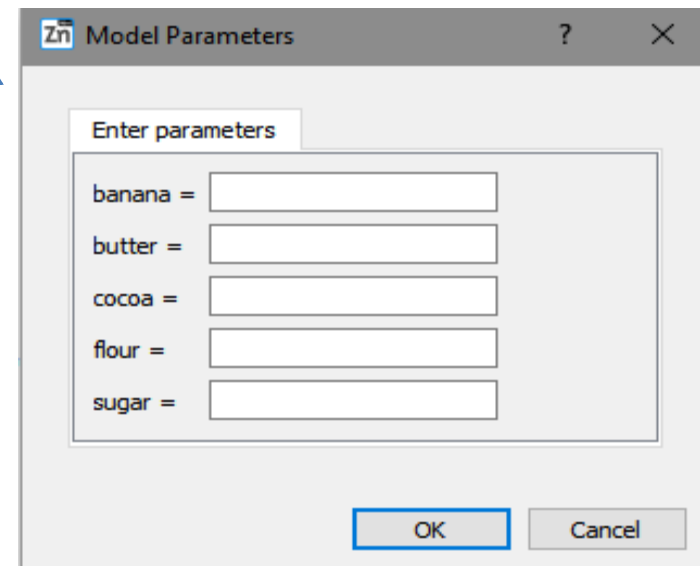
```
int: flour;    %no. grams of flour available  
int: banana;   %no. of bananas available  
int: sugar;    %no. grams of sugar available  
int: butter;   %no. grams of butter available  
int: cocoa;    %no. grams of cocoa available
```

%Parametros con Valor

```
int: flour = 4000;  
int: banana = 6;  
int: sugar = 2000;  
int: butter = 500;  
int: cocoa = 500;
```

**%Fichero datos1 ("datos1.dzn")**

```
flour = 4000;  
banana = 6;  
sugar = 2000;  
butter = 500;  
cocoa = 500;
```



## VARIABLES: Conjuntos

**set of** int|float|bool : <var-name> = <expresion> | {<exp<sub>1</sub>> <exp<sub>2</sub>>, ...<exp<sub>n</sub>>} ;

### *Ejemplos:*

int: P;           % P es un parámetro

set of int: Conjunto1 = 1 ..P;   % Conjunto1: variable de 1..P enteros

set of float: Conjunto2 = {2.5, 6.5, 10.5}; % Variable conjunto de reales

**Operaciones :**       Pertenencia (in, subset, superset),  
                          Union (union), Interseccion (inter),  
                          Diferencia (diff)  
                          Cardinalidad (card).

## Variable: Vectores

**array** [ $\langle \text{index-1} \rangle$ ,  $\langle \text{index-2} \rangle$ , .....,  $\langle \text{index-n} \rangle$ ] **of var** int|float|string|bool:  $\langle \text{var-name} \rangle$  ;

### *Ejemplos:*

int: N; %N es un parámetro

int: k=10; %k es un parámetro con valor indicado

array [1..N, 1..N] of var int: celda1; % celda1:array bi-dimensional de enteros

array [1..k] of var 1..100: celda2; % Array uni-dimensional, con valores 1..100

array [1..10, 1..5, 1..15] of var bool: celda3; % 3-dimensional de booleanos

Los vectores se pueden inicializarse :

celda1 = [ | 3, 5, ..... | 6, 7, ..... ]; % celda1: array bi-dimensional de N x N elementos

celda2 = [ 3, 5, 6, 7, ..... 76, 66 ]; % celda2: array unidimensional de 10 elementos

O adquirir sus valores de ficheros de datos externos.

**Ver diversos ejemplos en boletín sobre la definición, operativa e impresión de vectores!**



## Restricciones Especiales

*Diversos ejemplos en boletín y en documentación!*

<b>OR:</b>	<code>constraint s1 + d1 &lt;= s2 ∨ s2 + d2 &lt;= s1;</code>
<b>AND:</b>	<code>constraint s1 + d1 &lt;= s2 ∧ s2 + d2 &lt;= s1;</code>
<b>Condicional:</b>	<code>constraint if a &gt; b then c &gt; 10 else c &lt; 10 endif;    constraint if b &gt; c then d &gt; 10 endif;</code> <code>constraint if (s1 + d1 &lt;= s2 ∧ s2 + d2 &lt;= s1)</code> <code>          then (s1 + d1 &gt;= s3 ∨ s2 + d2 &gt;= s4) else c &lt; 10 endif;</code>
<b>Implicación:</b>	<code>constraint s1 + d1 &lt;= s2 -&gt; s2 + d2 &lt;= s1;    % Si</code> <code>constraint s1 + d1 &lt;= s2 &lt;- s2 + d2 &lt;= s1;    % Solo si</code> <code>constraint s1 + d1 &lt;= s2 &lt;-&gt; s2 + d2 &lt;= s1;    % Si y solo si</code>
<b>Negación:</b>	<code>constraint not ( s1 + d1 &lt;= s2 ∧ s2 + d2 &lt;= s1);</code>
<b>forall:</b>	<code>constraint forall (i,j in 1..3 where i &lt; j) (a[i] != a[j]);    % a[1] != a[2] ∧ a[1] != a[3] ∧ a[2] != a[3];</code>
<b>exists:</b>	<code>constraint exists (i,j in 1..3 where i &lt; j) (a[i] != a[j]);    % a[1] != a[2] ∨ a[1] != a[3] ∨ a[2] != a[3]</code>
<b>alldifferent:</b>	<code>include "alldifferent.mzn";                            % requiere incluir restricción global</code> <code>constraint alldifferent ([S,E,N,D,M,O,R,Y]);</code> <code>constraint alldifferent (Q);    % Los valores de las celdas del vector Q son todos diferentes</code> <code>constraint alldifferent (j in 1..N) ( Q[j] );        % Solo los primeros N valores son diferentes</code>

## Hay varios problemas resueltos en el boletín!!

Running sudoku.mzn

sudoku:

2 7 5 1 4 3 8 6 9

1 3 6 7 9 8 2 4 5

8 4 9 5 6 2 7 1 3

7 1 2 8 3 5 4 9 6

4 6 3 2 1 9 5 7 8

5 9 8 4 7 6 1 3 2

6 5 4 3 2 1 9 8 7

3 2 1 9 8 7 6 5 4

9 8 7 6 5 4 3 2 1

%Modelo de un Sudoku N x N

par int: S; %Parámetro pedido en la ejecución del modelo.

int: N = S\*S; %parametro, para usarlo como índice de la matriz

array [1..N, 1..N] of var 1..N: celda; % Sudoku, celda[i, j]

include "alldifferent.mzn";

% Todas las celdas en una fila son diferentes.

constraint forall (i in 1..N) ( alldifferent (j in 1..N) ( celda[i,j] ));

% Todas las celdas en una columna son diferentes.

constraint forall(j in 1..N) ( alldifferent (i in 1..N) ( celda[i,j] ));

% Todas las celdas en una submatriz son diferentes.

constraint forall (i,j in 1..S)

( alldifferent (p,q in 1..S) ( celda[S\*(i-1)+p, S\*(j-1)+q] ));

solve satisfy; %solo requerimos satisfabilidad

output [ "sudoku:\n" ] ++ [ show(celda[i,j]) ++ % Blancos separadores de submatrices

if j = N then if i mod S = 0 /\ i < N then "\n\n" else "\n" endif else

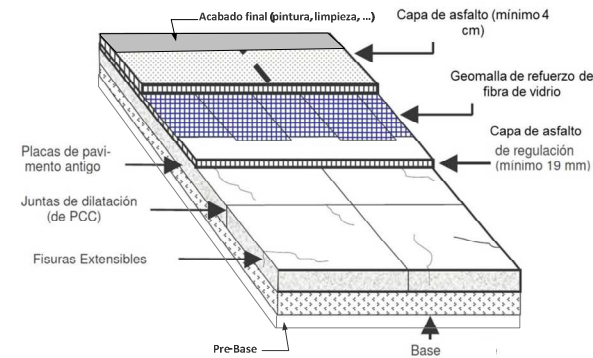
if j mod S = 0 then " " else " " endifendif | i,j in 1..N ];

## Práctica 3: CSP

### Tarea:

- Realizar el ejercicio propuesto (se necesitará para el día de la evaluación, en el que se planteará una breve ampliación o modificación)

{E1A, E1B, E2A, E2B,... , E9A, E9B}



### Calendario:

Sem	<u>LABORATORIO</u>	Evaluación
5-XII	CSP-MiniZinc	
12-XII	CSP-MiniZinc	
?		<b>P3: Eval: CSP-MiniZinc</b>

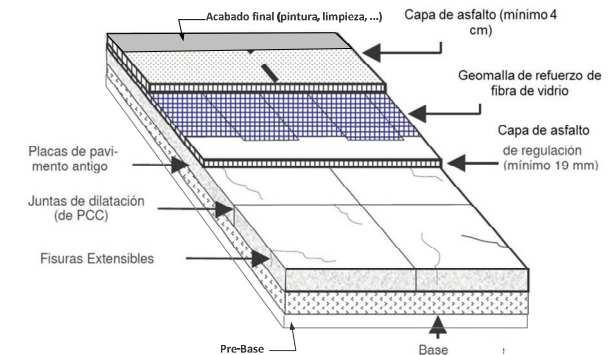
**Practica CSP (15%) P3**

## PROBLEMA

**Asfaltado de una Carretera:** 9 capas de distintos elementos {E1, E2, ... , E9}, cada una de ellas aplicadas de la forma (A:prensado y B:sinterizado):

**{E1A, E1B, E2A, E2B,... , E9A, E9B}**

La secuencia de aplicación de estas capas depende de condiciones ambientales, soporte base, uso, etc .



**Conjunto de Restricciones:** Restricciones para la aplicación de todas las capas {E1, E2, ... , E9}, en cada uno de los dos modos A/B.

### Ejemplo:

- La capa E2, aplicada de forma A (es decir E2A) no debe aplicarse junto a una capa E1, E5, E9 (aplicadas de modo A), ni a una capa E4 aplicada de modo B. Es decir, E2A no debe aplicarse junto a una capa E1A, E5A, E9A o E4B.
- Etc....

..... Más Posteriores ampliaciones