

# Planificación automática de horas de descargue para un puerto naval comercial

Angélica María Muñoz, Miguel Angel Reyes, Ricardo Andrés Pedreros

Septiembre 2021

## 1. Introducción

El presente documento tiene por objetivo presentar y explicar dos soluciones (con metodología distinta) planteadas por el equipo para el problema del cual hace alusión el título del documento y por lo tanto se pretende comprender y distinguir cuales son los beneficios logrados con cada uno de los planteamientos.

## 2. Planteamiento 1

### 2.1. Modelo

#### 2.1.1. Parámetros

- *Muelles*: Representa la cantidad de muelles disponibles para el proceso de descarga en el puerto, donde:

$$Muelles \in \mathbb{N}$$

- *Barcos*: Representa el número de barcos que llegarán en el día a planificar, donde:

$$Barcos \in \mathbb{N}$$

- *M\_codigoMuelles*: Arreglo que contiene el código único de cada muelle, en él está la información de todos los muelles, no solo de los que están en el puerto determinado. En donde:

$$M\_codigoMuelles_i \in \mathbb{N} \\ i \in (1..length(M\_codigoMuelles))$$

- *B\_codigoBarcos*: Arreglo que contiene el código único de cada barco, en él está la información de todos los barcos, no solo de los que planean llegar por día. En donde:

$$\begin{aligned} B\_codigoBarcos_i &\in \mathbb{N} \\ i &\in (1..length(M\_codigoBarcos)) \end{aligned}$$

- *B\_tpEspera*: Arreglo que contiene el tiempo máximo de espera entre la llegada y el inicio de la descarga de cada barco. En cada *B\_tpEspera<sub>i</sub>* hay un número que representa dicho tiempo en slots, cada 1 slot equivale a 5 minutos. Este arreglo contiene la información de todos los barcos, no solo de los que planean llegar por día. En donde:

$$\begin{aligned} B\_tpEspera_i &\in \mathbb{N} \\ i &\in (1..length(M\_tpEspera)) \end{aligned}$$

- *B\_tpdescarga*: Arreglo que contiene el tiempo que demora cada barco en su proceso de descarga. En cada *B\_tpdescarga<sub>i</sub>* hay un número que representa dicho tiempo en slots, cada 1 slot equivale a 5 minutos. Este arreglo contiene la información de todos los barcos, no solo de los que planean llegar por día. En donde:

$$\begin{aligned} B\_tpdescarga_i &\in \mathbb{N} \\ i &\in (1..length(M\_tpdescarga)) \end{aligned}$$

- *B\_estadost*: Arreglo que contiene las condiciones climáticas que acepta cada barco. En cada posición *B\_estadost<sub>i</sub>* hay un arreglo de tres elementos que representa los estados del clima, así: [seco,húmedo,lluvioso]. Por lo anterior, cada una de esas posiciones contiene un valor booleano (0 o 1), el cual muestra si el barco *i* puede (1) o no puede (0) estar en el muelle esperando o descargando con dicho clima. En este arreglo se encuentra la información de todos los barcos, no solo de los que planean llegar por día. En donde:

$$\begin{aligned} B\_estadost_{ij} &\in \{0,1\} \\ i &\in (1..length(M\_estadost)), j \in [1,3] \end{aligned}$$

- *B\_estadosm*: Arreglo que contiene las condiciones de la marea que acepta cada barco. En cada posición *B\_estadosm<sub>i</sub>* hay un arreglo de tres elementos que representa los estados del clima, así: [baja,media,alta]. Por lo anterior, cada una de esas posiciones contiene un valor booleano (0 o 1), el cual muestra si el barco *i* puede (1) o no puede (0) estar en el muelle esperando o descargando con dicha marea. En este arreglo se encuentra la información de todos los barcos, no solo de los que planean llegar por día. En donde:

$$B\_estadost_{ij} \in \{0, 1\}$$

$$i \in (1..length(M\_estadost)), j \in [1, 3]$$

- *P\\_estadost*: Arreglo que contiene las condiciones del clima durante el día a planear en el puerto. Las unidades de tiempo son slots, donde 1 slot es equivalente a 5 minutos. Cada  $P\_estadost_i$  puede contener un 1, 2 o 3, los que representan el clima seco, húmedo y lluvioso respectivamente. Así:

$$P\_estadost_i \in \{1, 3\}$$

$$i \in [1, 288]$$

- *P\\_estadostm*: Arreglo que contiene las condiciones de la marea durante el día a planear en el puerto. Las unidades de tiempo son slots, donde 1 slot es equivalente a 5 minutos. Cada  $P\_estadostm_i$  puede contener un 4, 5 o 6, los que representan la marea baja, media y alta respectivamente. Así:

$$P\_estadostm_i \in \{4, 6\}$$

$$i \in [1, 288]$$

- *HoraLlegada*: Arreglo que contiene información sobre los barcos que tienen una hora de llegada específica. Partiendo de lo anterior, cada  $horaLlegada_i$  tiene un arreglo de dos elementos, en donde  $horaLlegada_{i1}$  es el número del barco y  $horaLlegada_{i2}$  es el tiempo en el que dicho barco debe llegar al puerto. Este tiempo se representa con slots, en donde 1 slot es igual a 5 minutos. Así:

$$horaLlegada_{i1} \in [1..barcos]$$

$$horaLlegada_{i2} \in [1..length(P\_estadost)]$$

$$i \in [1..barcos], j \in [1, 2]$$

- *HorasHabiles*: Arreglo que contiene la información sobre las horas en que está habilitado cada muelle durante el día, en él está la información de todos los muelles, no solo de los que están en el puerto determinado. En cada posición  $horasHabiles_i$  hay un arreglo de dos elementos en donde  $horasHabiles_{i1}$  contiene la hora de apertura del muelle y  $horasHabiles_{i2}$  la hora en que cierra el muelle. Así:

$$horasHabiles_{ij} \in [1..length(P\_estadost)]$$

$$i \in (1..length(horasHabiles)), j \in [1, 2]$$

- *PreasignacionHoraLlegada*: Es un valor booleano (0 o 1) que representa si restringir la hora de llegada de los barcos es de manera obligatoria (1) o si solo es preferible hacerlo, pero no necesario (0). En donde:

$$PreasignacionHoraLlegada \in \{0, 1\}$$

### 2.1.2. Redefinición de algunos parámetros

Dado que algunas entradas no estaban condicionadas a la cantidad de barcos que llegarían cada día al puerto, se crearon nuevos arreglos, los cuales solo contienen la información necesaria respecto a barcos. Así:

- *shipsID*: Arreglo que contiene los mismos elementos de *B\_codigoBarcos*, pero solo con la cantidad de barcos que llegarán ese día. En donde:

$$\begin{aligned} shipsID_i &\in \mathbb{N} \\ i &\in [1..barcos] \end{aligned}$$

- *shipsWaitTime*: Arreglo que contiene los mismos elementos de *B\_tpEspera*, pero solo con la cantidad de barcos que llegarán ese día. En donde:

$$\begin{aligned} shipsWaitTime_i &\in \mathbb{N} \\ i &\in [1..barcos] \end{aligned}$$

- *shipsUnloadTime*: Arreglo que contiene los mismos elementos de *B\_tpdescarga*, pero solo con la cantidad de barcos que llegarán ese día. En donde:

$$\begin{aligned} shipsUnloadTime_i &\in \mathbb{N} \\ i &\in [1..barcos] \end{aligned}$$

- *shipsWeather*: Arreglo de dos dimensiones que contiene los mismos elementos de *B\_estadost*, pero solo con la cantidad de barcos que llegarán ese día. En donde:

$$\begin{aligned} shipsWeather_{ij} &\in \{0, 1\} \\ i &\in [1..barcos], j \in [1..3] \end{aligned}$$

- *shipsTideState*: Arreglo de dos dimensiones que contiene los mismos elementos de *B\_estadosm*, pero solo con la cantidad de barcos que llegarán ese día. En donde:

$$\begin{aligned} shipsTideState_{ij} &\in \{0, 1\} \\ i &\in [1..barcos], j \in [1..3] \end{aligned}$$

De igual manera, partiendo de que algunas entradas no tienen en cuenta los muelles del puerto determinado, sino otros más. Se creó un set de números, en donde se limita esta información solo a la necesaria, de la siguiente manera:

- *docksID*: Set de enteros que contiene los mismos elementos de *M\_codigoMuelles*, pero solo con la cantidad de muelles existentes en el puerto. En donde:

$$\begin{aligned} docksID_i &\in \mathbb{N} \\ i &\in [1..muelles] \end{aligned}$$

### 2.1.3. Variables

- *docks*: Arreglo variable de tamaño *barcos* que se encargará de representar respectivamente para cada barco el muelle correspondiente al que tendrá que llegar. En donde:

$$\begin{aligned} docks_i &\in docksID \\ i &\in \mathbb{N} \end{aligned}$$

- *arrivalTime*: Arreglo variable de tamaño *barcos* que se encargará de representar respectivamente para cada barco el tiempo en el cual deberá llegar. En donde:

$$\begin{aligned} arrivalTime_i &\in \{1, length(P\_estadost)\} \\ i &\in \mathbb{N} \end{aligned}$$

- *unloadStartTime*: Arreglo variable de tamaño *barcos* que se encargará de representar respectivamente para cada barco el tiempo en el cual deberá comenzar la descarga. En donde:

$$\begin{aligned} unloadStartTime_i &\in \{1, length(P\_estadost)\} \\ i &\in \mathbb{N} \end{aligned}$$

### 2.1.4. Restricciones

Al realizar el análisis de las restricciones, se llegó a tres que se explicarán a continuación:

- En primera medida se estableció la condición que limita a que si dos barcos van a estar en el mismo puerto, no pueden hacerlo al mismo tiempo, o tener cruces, y esto se logró mediante el siguiente razonamiento:

Si dos barcos (A y B) van a llegar al mismo puerto, y si A llega antes que B, entonces el momento en que A termine de descargar va a ser menor al tiempo de llegada de B; planteandolo con base en las variables y de manera matemática, se vería así:

Sean  $i$  y  $j$  posiciones que representan la posición de dos barcos ( $i \neq j$ ) donde  $i, j \in [1..barcos]$ , se va a considerar lo siguiente:

$$\begin{aligned} &\text{Si } docks_i = docks_j \wedge arrivalTime_i < arrivalTime_j \text{ entonces:} \\ &\forall i, j \Rightarrow unloadStartTime_i + shipsUnloadTime_i \leq arrivalTime_j \end{aligned}$$

- La siguiente condición establecida se centró en el evitar la toma de valores incoherentes, a continuación se explicará:

Si un barco A llega en un tiempo  $t_0$  y comienza a descargar en un tiempo  $t_d$ , entonces,  $t_0$  debe ser menor a  $t_d$ , observándolo de manera matemática sería de la siguiente manera:

Sea  $i$  una posición que representa a un barco, entonces:

$$\forall i \Rightarrow arrivalTime_i \leq unloadStartTime_i$$

- Otra condición determinada se centró en validar la duración del tiempo de espera de cada barco, es decir, el tiempo en que tardan desde que llegan hasta que comienzan a descargar, esto se pensó de la siguiente manera:

Si un barco A llega en un determinado tiempo  $t_0$ , y empieza a descargar en un tiempo  $t_d$  entonces, la diferencia de esos tiempos no debe ser mayor al tiempo de espera indicado en los datos dados; ahora se mostrará de manera matemática de esta restricción:

Sea  $i$  una posición que representa a un barco, entonces:

$$\forall i \Rightarrow unloadStartTime_i - arrivalTime_i \leq shipsWaitTime_i$$

- Adicionalmente, una restricción planteada fue centrada en verificar tanto las condiciones climáticas como de marea mediante el siguiente razonamiento:

Si un barco A tiene unos climas permitidos C, unos niveles de marea permitidos M, y llega al puerto en un tiempo  $t_0$  y termina de descargar en un tiempo  $t_f$ , donde  $[t_0, t_f]$  es el intervalo de tiempo que el barco permanece en el puerto, entonces durante este intervalo de tiempo el clima en el puerto debe ser igual a alguno de los climas permitidos C, y el nivel de marea igual a algún nivel de marea permitido M; esto matemáticamente sería:

Sea  $i$  una posición que representa un barco, sea

$[arrivalTime_i \dots unloadStartTime_i + shipsUnloadTime_i]$  el intervalo de tiempo  $T_i$  que un barco  $i$  permanece en el muelle  $docks_i$ , y sea  $t_j$  el elemento  $j$ -ésimo de  $T_i$ , entonces:

\*Caso Clima

$$\forall i, t_j \Rightarrow P\_estadost_{t_j} = 1 * shipsWeather_{i,1}$$

$\vee$

$$P\_estadost_{t_j} = 2 * shipsWeather_{i,2}$$

$\vee$

$$P\_estadost_{t_j} = 3 * shipsWeather_{i,3}$$

\*Caso Marea

$$\forall i, t_j \Rightarrow P\_estadost_{t_j} = 4 * shipsTideState_{i,1}$$

$\vee$

$$P\_estadost_{t_j} = 5 * shipsTideState_{i,2}$$

$\vee$

$$P\_estadost_{t_j} = 6 * shipsTideState_{i,3}$$

- También se planteó una restricción que hace referencia a los horarios activos que pueden tener establecidos los muelles, es decir, que solo en ese tiempo el muelle puede albergar barcos, de manera matemática se puede indicar de la siguiente manera:

Sea  $docks_i$  el muelle al que va a llegar un barco  $i$ , sea  $docksID_j$  el ID del muelle  $j$ -ésimo, y si  $docks_i = docksID_j$  entonces:

$$\begin{aligned} \forall i, j \Rightarrow arrivalTime_i \geq horasHabiles_{j,1} \\ \wedge \\ unloadStartTime_i + shipsUnloadTime_i - 1 \leq horasHabiles_{j,2} \end{aligned}$$

- La condición que hace referencia a la preasignación de la hora de llegada de un barco fue interpretada de una manera muy sencilla de la siguiente manera:

Si un barco A llega en un tiempo  $t_a$  y sea  $preassign_a$  la hora preasignada a la cual debe llegar el barco A, entonces se debe cumplir que  $t_a = preassign_a$ , y matemáticamente la podemos expresar así:

Sea  $horaLlegada_i, 1$  el índice que hace referencia al barco y  $horaLlegada_i, 2$  la hora a la cual se desea que llegue, entonces:

$$\forall i \Rightarrow arrivalTime_{horaLlegada_i,1} = horaLlegada_i,2$$

- Finalmente para el manejo de la preasignación de manera débil, es decir, si existe alguna solución con el barco llegando a esa hora, entonces que se muestre, de lo contrario, que muestre otra que no cumpla esa condición, se utilizó **reificación**, donde planteamos para cada una de las cinco restricciones anteriormente explicadas una variable booleana, las cuales, respectivamente son: *noOverlap*, *maxWait* (incluye a la segunda y tercer restricción), *climatology*, *arriShips* y *scheDocks*, y adicionalmente la variable entera a maximizar que es *REIF*.

Para “obligar” a que las restricciones se cumplan, se realizó una suma donde estaban involucradas las variables *noOverlap*, *maxWait*, *climatology* y *scheDocks*, la cual debía dar como resultado 4 ya que es la única manera que se garantiza que las 4 variables toman valor de True (o valor numérico de 1).

Y para concluir se realiza la suma de todas las 5 variables y se le asigna a la variable REIF, la cual será maximizada por el solver.

## 2.2. Pruebas

Inicialmente, en las pruebas, se usó un ejemplo sencillo, fácil de resolver manualmente, y corto, donde se verificó que retornara todas las soluciones posibles y se analizó el comportamiento de las validación básicas (climatológicas, de marea, no solapamiento y de datos coherentes):

```

%General:
muelles = 1;
barcos = 2;

%Barcos
B_codigoBarcos = [23,78];

%Tiempo máximo de espera de un barco
B_tpEspera = [3,1];

%Condiciones de clima para cada barco. Se asume un ordenamiento [seco,húmedo,lluvia]
B_estadost = [[1,0,0|1,1,0]];

%Tiempo de descarga de un barco
B_tpdescarga = [0,2];

%Estados de mareas para un barco dado un ordenamiento [baja,media,alta]
B_estadosm = [[0,0,1|0,0,1]];

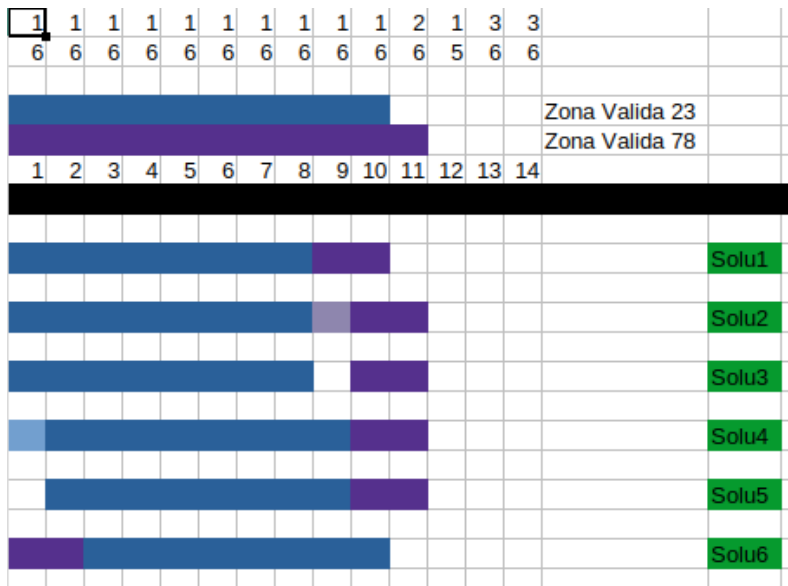
%Muelles
M_codigoMuelles = [45];

%Estados del tiempo en el puerto. 1 = seco, 2 = húmedo, 3 = lluvia
P_estadost=[1,1,1,1,1,1,1,1,1,1,2,1,3,3];
%      A,B,C,D,E,F,G,H,I,J,K,L,M,N

%Estados de la marea cada 5 minutos en el puerto. 4 = baja, 5 = media, 6 = alta
P_estadosm=[6,6,6,6,6,6,6,6,6,6,6,5,6,6];
%      A,B,C,D,E,F,G,H,I,J,K,L,M,N

```

*Datos de prueba trivial*



*Análisis manual de las soluciones*



```

Running RBasicas.mzn with Prueba1.dzn
Barcos      :[23, 78]
Muelles     :[45, 45]
Hora de llegada :[1, 9]
T max. de espera:[3, 1]
Hora de descarga:[1, 9]
Tiempo descarga :[8, 2]
-----
Barcos      :[23, 78]
Muelles     :[45, 45]
Hora de llegada :[1, 9]
T max. de espera:[3, 1]
Hora de descarga:[1, 10]
Tiempo descarga :[8, 2]
-----
Barcos      :[23, 78]
Muelles     :[45, 45]
Hora de llegada :[1, 10]
T max. de espera:[3, 1]
Hora de descarga:[1, 10]
Tiempo descarga :[8, 2]
-----
Barcos      :[23, 78]
Muelles     :[45, 45]
Hora de llegada :[1, 10]
T max. de espera:[3, 1]
Hora de descarga:[2, 10]
Tiempo descarga :[8, 2]
-----
Barcos      :[23, 78]
Muelles     :[45, 45]
Hora de llegada :[2, 10]
T max. de espera:[3, 1]
Hora de descarga:[2, 10]
Tiempo descarga :[8, 2]
-----
Barcos      :[23, 78]
Muelles     :[45, 45]
Hora de llegada :[3, 1]
T max. de espera:[3, 1]
Hora de descarga:[3, 1]
Tiempo descarga :[8, 2]
-----

```

*Soluciones encontradas por el solver*

De la misma forma se agregaron las condiciones adicionales para verificar su correcto funcionamiento, y se logra concluir que, para las pruebas realizadas, el modelo lograba encontrar, de manera completa y sin ningún problema, todas las soluciones.

Adicionalmente, se tomó el ejemplo *datosPruebaMinizinc2.dzn* para realizar un breve análisis con respecto a la eficiencia de dos solvers: **Chuffed** y **Gecode** y poder observar el comportamiento usando datos relativamente complejos, y se logró corroborar que *Chuffed* tiene una eficiencia medianamente mejor, ya que logra encontrar la(s) solución(es) un 10 % más rápido, tanto utilizando la preasignación fuerte, como la preasignación débil.

## 2.3. Conclusiones planteamiento 1.

El uso de reificaciones es una metodología bastante útil y eficiente a la hora de hacer restricciones y en concreto a la hora de maximizar o minimizar la cantidad que se requiere de estas. Pues, como se puede observar en este planteamiento se puede ver una cantidad de restricciones reducida a comparación de si se hubiera usado otra estrategia. Al mismo tiempo, gracias a utilizar esto el tamaño del

árbol no crece de una manera innecesaria con restricciones que podrían empeorar la situación. Por otro lado, se puede observar que el uso de diferentes solvers no influye de una manera tan considerable (10% más rápido con chuffed) a este planteamiento ya que el tamaño del árbol no debe ser muy permisivo a la hora de podar con las diferentes estrategias de los solvers.

Por lo que se puede concluir que con este planteamiento por su sencillez, las diferentes estrategias y metodologías aplicadas en este dan como resultado a un modelo y una solución rápida ya que no se genera un árbol demasiado grande.

### 3. Planteamiento 2.

#### 3.1. Idea general para resolver el problema.

Lo que se hará para resolver el problema será tener un arreglo que sus columnas representarán los slots de tiempo en el puerto y sus filas cada muelle, para el cual se asignará respectivamente los barcos, teniendo en cuenta las demás restricciones de disponibilidad para esto.

Después de realizar esto, se tendrá un arreglo para representar el uso del tiempo en los barcos, es decir las columnas representarán los slots de tiempo y las filas cada barco, para el cual las posiciones tomadas por un barco en el arreglo de los muelles se copiarán al arreglo de los barcos y después se tendrá en cuenta el tiempo de espera de cada barco (de haber necesidad). Además se aplicarán las restricciones debidas respecto al clima, la marea y demás.

#### 3.2. Modelo.

##### 3.2.1. Parámetros.

- Sea *muelles* el número de muelles que se tendrán disponibles en la planeación del puerto.
- Sea *barcos* el número de barcos que se llegarán en el día para la planeación del puerto.
- Sea *B\_codigoBarcos* un arreglo donde  $B\_codigoBarcos_i \in \mathbf{N}$  y cada uno de estos representará el código correspondiente de cada barco.
- Sea *B\_tpEspera* un arreglo donde  $B\_tpEspera \in \mathbf{N}$  y cada uno de estos representará el tiempo de espera que tiene disponible cada barco respectivamente (cada índice corresponde).
- Sea *B\_tpdescarga* un arreglo donde  $B\_tpdescarga \in \mathbf{N}$  y cada uno de estos representará el tiempo que se toma con cada barco para descargar su contenido en el muelle (cada índice corresponde).
- Sea *B\_estadost* un arreglo de dos dimensiones  $m \times 3$  donde el orden que representarán las columnas será [*seco*, *humedo*, *lluvia*] y cada fila será cada

barco respectivamente, esto para poder decir en que climas el barco puede estar, con un 1 de ser posible y un 0 de lo contrario.

- Sea  $B\_estadosm$  un arreglo de dos dimensiones  $m \times 3$  donde el orden que representarán las columnas será  $[baja, media, alta]$  y cada fila será cada barco respectivamente, esto para poder decir en que mareas el barco puede estar, con un 1 de ser posible y un 0 de lo contrario.
- Sea  $M\_codigoMuelles$  un arreglo donde  $M\_codigoMuelles_i \in \mathbf{N}$  y cada uno de estos representará el código correspondiente de cada muelle.
- Sea  $P\_estadost$  un arreglo donde  $P\_estadost_i \in [1, 3]$  y cada elemento representará el clima que se tiene estipulado para ese slot de tiempo.
- Sea  $P\_estadosm$  un arreglo donde  $P\_estadosm_i \in [1, 3]$  y cada elemento representará la marea que se tiene estipulada para ese slot de tiempo.
- Sea  $horaFlexible$  un arreglo de dos dimensiones  $n \times 2$  que nos dirá si se tiene flexibilidad con una restricción de llegada o no de los barcos al puerto donde la primera columna representará el barco en cuestión y la segunda columna si su restricción sera flexible, 0 para debil y 1 para fuerte.
- Sea  $horaLlegada$  un arreglo de dos dimensiones  $n \times 2$  que nos dirá la restricción de llegada al puerto que tendrá cada barco, donde la primera columna nos dirá el barco en cuestión y la segunda el slot de llegada.
- Sea  $horasHabiles$  un arreglo de dos dimensiones  $n \times 2$  que nos dirá la restricción de disponibilidad del muelle, donde la fila representará el muelle en cuestión, la primera columna el slot de inicio y la segunda columna el slot final del muelle.

### 3.2.2. Variables.

Para continuar con el proceso, tendremos en cuenta las siguientes funciones:

- Sea  $length(n)$  una función que recibe un arreglo y entrega su tamaño.
- Sea  $round(n)$  una funcion que recibe un número y lo redondea.

Teniendo esto en cuenta, las variables del modelo son las siguientes:

- Sea  $P\_estadostUsar$  un arreglo de dos dimensiones donde  $i \in [1, round(length(horasHabiles)/2)]$  y  $j \in [1, length(P\_estadost)]$  y cada uno de sus elementos  $P\_estadostUsar_{i,j} \in [0, 3]$ . Esto será utilizado para descartar los muelles que no se utilizarán respecto al clima.
- Sea  $P\_estadostFinal$  un arreglo de dos dimensiones donde  $i \in [1, muelles]$  y  $j \in [1, length(P\_estadost)]$  y cada uno de sus elementos  $P\_estadostFinal_{i,j} \in [0, 3]$ . Esto será utilizado para descartar junto a  $P\_estadostUsar$  los tiempos en el muelle que no estará disponible.

- Sea *usoMuelle* un arreglo de dos dimensiones donde  $i \in [1, muelles]$  y  $j \in [1, length(P\_estadost)]$  y cada uno de sus elementos  $usoMuelle_{i,j} \in [0, barcos]$ . Esto será con el objetivo de marcar el número correspondiente del barco en el slot que ocupará el barco para su descarga, cabe resaltar que el muelle será representado por las filas y los slots de tiempo por las columnas, además el slot donde su valor sea 0 querra decir que ningun barco ocupará este tiempo.
- Sea *tiempoPorBarco* un arreglo de dos dimensiones donde  $i \in [1, barcos]$  y  $j \in [1, length(P\_estadost)]$  y cada uno de sus elementos  $tiempoPorBarco_{i,j} \in [0, barcos]$ . Esto será con el objetivo de marcar el uso de slots de tiempo de cada barco correspondiente mente (tiempo de descarga + tiempo de espera) para saber su estadía total en el puerto. Cabe resaltar que los barcos se verán reflejados en las filas y los slots de tiempo en las columnas.
- Sea *docks* un arreglo de tamaño *barcos* que tendrá por objetivo presentar la respuesta respectivamente para cada barco y su muelle.
- Sea *arrivalTime* un arreglo de tamaño *barcos* que tendrá por objetivo presentar la respuesta respectivamente para cada barco y su tiempo en el que llega.
- Sea *unloadStartTime* un arreglo de tamaño *barcos* que tendrá por objetivo presentar la respuesta respectivamente para cada barco y su tiempo en el que empieza la descarga.

### 3.2.3. Restricciones.

Para las siguientes restricciones tendremos en cuenta los dos siguientes predicados: *count*( $x, y, c$ ) donde  $c$  será el número de ocurrencias de  $y$  en  $x$  y el segundo predicado que es *abs*( $n$ ) el cual recibe cualquier número y entrega su valor absoluto.

- El objetivo de esta restricción es verificar la máxima distancia (en el arreglo respectivo a muelles) que puede tener un slot de tiempo usado por un barco entre todos sus demás slots de tiempo en el muelle (es decir, que esten juntos)

$$\begin{aligned} & \forall j \in [1, barcos] \wedge m \in [1, muelles] | \dots \\ & \dots ( \sum_{i,n=1}^{length(P\_estadost)} usoMuelle_{m,i}, donde \dots \\ & \dots usoMuelle_{m,i} = j \wedge usoMuelle_{m,n} = j \wedge i \neq n \wedge abs(i-n) > B\_tpdescarga_j ) = 0 \end{aligned}$$

- El objetivo de la siguiente restricción es verificar la cantidad máxima de slots usados que debe tener un barco en el muelle.

$$\forall j \in [1, barcos] \wedge m \in [1, muelles] | \dots$$

$$\begin{aligned}
& \dots \left( \sum_{i=1}^{length(P\_estadost)} 1, \text{donde} \dots \right. \\
& \dots \text{usoMuelle}_{m,i} = j) = B\_tpdescarga[j]) \oplus \dots \\
& \dots \left( \sum_{i=1}^{length(P\_estadost)} 1, \text{donde} \dots \right. \\
& \dots \text{usoMuelle}_{m,i} = j) = 0)
\end{aligned}$$

- El objetivo de la siguiente restricción es verificar la cantidad máxima de apariciones de un barco en todo el puerto.

$$\forall i \in [1, \text{barcos}] | \text{count}(\text{usoMuelle}, i, B\_tpdescarga[i])$$

- Con la siguiente restricción se tiene por objetivo, asignar con 0 los slots de tiempo en que el muelle no pueda ser usado.

$$\forall i \in [1, \text{muelles}] \wedge j \in [1, length(P\_estadost) \wedge P\_estadostFinal_{i,j} = 0 | \text{usoMuelle}_{i,j} = 0]$$

- La siguiente restricción tiene por objetivo copiar los slots de tiempo utilizados en el muelle por cada barco, en los slots de tiempo de cada barco.

$$\forall i \in [i, \text{muelles}] \wedge j \in [1, 1..length(P\_estadost)] \wedge \text{usoMuelle}_{i,j} > 0 | \dots$$

$$\dots \text{tiempoPorBarco}_{\text{usoMuelle}_{i,j},j} = \text{usoMuelle}_{i,j}$$

- Con la siguiente restricción se hará que el arreglo que contiene los slots de tiempo de cada barco, se asegure que para cada barco solo aparezcan como valor distinto de cero los slots en donde él estuvo en el puerto.

$$\forall i \in [1, \text{barcos}] \wedge j \in [1, length(P\_estadost)] \wedge \text{tiempoPorBarco}_{i,j} \neq i | \text{tiempoPorBarco}_{i,j} = 0$$

- La siguiente restricción tiene por objetivo que la cantidad mínima de uso en los slots de tiempo para cada barco sea igual o mayor a su tiempo de descarga.

$$\begin{aligned}
& \forall i \in [1, \text{barcos}] | \left( \sum_{j=1}^{length(P\_estadost)} 1, \text{donde} \dots \right. \\
& \dots \text{tiempoPorBarco}_{i,j} = i) \geq B\_tpdescarga_i
\end{aligned}$$

- La siguiente restricción tiene por objetivo que la cantidad maxima de uso en los slots de tiempo para cada barco sea igual o menor a la suma de su tiempo de descarga y su tiempo de espera.

$$\begin{aligned}
& \forall i \in [1, \text{barcos}] | \left( \sum_{j=1}^{length(P\_estadost)} 1, \text{donde} \dots \right. \\
& \dots \text{tiempoPorBarco}_{i,j} = i) \leq (B\_tpEspera_i + B\_tpdescarga_i)
\end{aligned}$$

- El objetivo de esta restricción es verificar la máxima distancia (en el arreglo respectivo a barcos) que puede tener un slot de tiempo usado por un barco entre todos sus demás slots de tiempo en el muelle (es decir, que esten juntos). Cabe resaltar que la distancia en este arreglo puede ser mayor a la que existe en los muelles, pues aquí se tiene añadido el tiempo de espera.

$$\forall j \in [1, \text{barcos}] | \dots$$

$$\dots \left( \sum_{i,n=1}^{\text{length}(P\_estadost)} \text{tiempoPorBarco}_{j,i}, \text{ donde } \dots \right.$$

$$\dots \text{tiempoPorBarco}_{j,i} = j \wedge \text{tiempoPorBarco}_{j,n} = j \wedge i \neq n \wedge \text{abs}(i - n) \dots$$

$$\dots \geq (B\_tpdescarga_j + B\_tpEspera_j) = 0$$

- La siguiente restricción tiene por objetivo determinar si un barco tiene que llegar en x slot de tiempo o preferentemente debe hacerlo.

$$\forall i \in [1, \text{round}(\text{length}(\text{horaLlegada})/2)] | \dots$$

$$\dots (\text{horaFlexible}_{i,2} = 1) - > \dots$$

$$\dots ((i = 1) - > (\text{tiempoPorBarco}_{\text{horaLlegada}_{i,1}, \text{horaLlegada}_{i,2}} = \text{horaLlegada}_{i,1}) \dots$$

$$\dots \oplus (\text{tiempoPorBarco}_{\text{horaLlegada}_{i,1}, \text{horaLlegada}_{i,2}} = \text{horaLlegada}_{i,1} \wedge \dots$$

$$\dots \text{tiempoPorBarco}_{\text{horaLlegada}_{i,1}, \text{horaLlegada}_{i,2}-1} \neq \text{horaLlegada}_{i,1}) \dots$$

$$\dots \oplus ((i = 1) - > ((\text{tiempoPorBarco}_{\text{horaLlegada}_{i,1}, \text{horaLlegada}_{i,2}} = \text{horaLlegada}_{i,1}) \dots$$

$$\dots \oplus (\text{tiempoPorBarco}_{\text{horaLlegada}_{i,1}, \text{horaLlegada}_{i,2}} \neq \text{horaLlegada}_{i,1})) \oplus \dots$$

$$\dots ((\text{tiempoPorBarco}_{\text{horaLlegada}_{i,1}, \text{horaLlegada}_{i,2}} = \text{horaLlegada}_{i,1} \dots$$

$$\dots \wedge \text{tiempoPorBarco}_{\text{horaLlegada}_{i,1}, \text{horaLlegada}_{i,2}-1} \neq \text{horaLlegada}_{i,1}) \oplus \dots$$

$$\dots (\text{tiempoPorBarco}_{\text{horaLlegada}_{i,1}, \text{horaLlegada}_{i,2}} \neq \text{horaLlegada}_{i,1}))$$

- La siguiente restricción tiene por objetivo revisar que se cumplan las condiciones climaticas para cada barco.

$$\forall i \in [1, \text{barcos}] \wedge j \in [1, \text{length}(P\_estadost)] \wedge \text{tiempoPorBarco}_{i,j} = i | \dots$$

$$\dots ((P\_estadost_j = 1 \wedge B\_estadost_{i,1} = 1) - > (\text{true})) \oplus \dots$$

$$\dots ((P\_estadost_j = 2 \wedge B\_estadost_{i,2} = 1) - > (\text{true})) \oplus \dots$$

$$\dots ((P\_estadost_j = 3 \wedge B\_estadost_{i,3} = 1) - > (\text{true})) \oplus (\text{false})$$

- La siguiente restricción tiene por objetivo revisar que se cumplan las condiciones de marea para cada barco.

$$\begin{aligned} & \forall i \in [1, \text{barcos}] \wedge j \in [1, \text{length}(P\_estadost)] \wedge \text{tiempoPorBarco}_{i,j} = i | \dots \\ & \dots ((P\_estadost_m_j = 4 \wedge B\_estadost_{i,1} = 1) \rightarrow (true)) \oplus \dots \\ & \dots ((P\_estadost_m_j = 5 \wedge B\_estadost_{i,2} = 1) \rightarrow (true)) \oplus \dots \\ & \dots ((P\_estadost_m_j = 6 \wedge B\_estadost_{i,3} = 1) \rightarrow (true)) \oplus (false) \end{aligned}$$

- Con la siguiente restricción se pensá quitar los muelles que no estaran disponibles.

$$\begin{aligned} & \forall i \in [1, \text{round}(\text{length}(\text{horasHabiles})/2)] | \forall m \in [1, \text{round}(\text{length}(\text{horasHabiles})/2)] \dots \\ & \dots \wedge n \in [1, \text{length}(P\_estadost)] \wedge m = i | ((n \leq \text{horasHabiles}_{i,2} \wedge n \geq \text{horasHabiles}_{i,1}) \dots \\ & \dots \rightarrow (P\_estadostUsar_{m,n} = P\_estadost_n)) \oplus (P\_estadostUsar_{m,n} = 0) \end{aligned}$$

- Con la siguiente restricción se pensá quitar los slots de tiempo que cada muelle no tendra disponible.

$$\forall i \in [1, \text{muelles}] \wedge j \in [1, \text{length}(P\_estadost)] | P\_estadostFinal_{i,j} = P\_estadostUsar_{i,j}$$

Las siguientes restricciones tienen por objetivo, organizar la respuesta, para su facil interpretación. Para estas restricciones tendremos en cuenta los siguientes dos predicados:  $\min(x)$  el cual regresa el valor mínimo de una lista de números y  $\max(x)$  el cual regresa el valor máximo de una lista de números.

- $\forall i \in [1, \text{barcos}] \wedge j \in [1, \text{length}(P\_estadost)] \wedge \text{tiempoPorBarco}_{i,j} = i | \text{respuesta}_{i,1} = B\_codigoBarcos_i$
- $\forall i \in [1, \text{barcos}] \wedge j \in [1, \text{length}(P\_estadost)] \wedge m \in [1, \text{muelles}] \wedge \text{usoMuelle}_{m,j} = i | \text{respuesta}_{i,2} = M\_codigoMuelles_m$
- $\forall i \in [1, \text{barcos}] | \text{respuesta}_{i,3} = \min([j | j \in [1, \text{length}(P\_estadost)] \wedge \text{tiempoPorBarco}_{i,j} = i])$
- $\forall i \in [1, \text{barcos}] | \text{respuesta}_{i,4} = \min([j | m \in [1, \text{muelles}] \wedge j \in [1, \text{length}(P\_estadost)] \wedge \text{usoMuelle}_{m,j} = i])$
- $\forall i \in [1, \text{barcos}] | \text{respuesta}_{i,5} = \max([j | m \in [1, \text{muelles}] \wedge j \in [1, \text{length}(P\_estadost)] \wedge \text{usoMuelle}_{m,j} = i])$

### 3.2.4. Pruebas.

Para analizar los resultados se hara uso de dos solvers: Gecode 6.3.0 y Chuffed 0.10.4.

Como se puede observar en la siguiente imagen que, con una entrada de 2 muelles, 4 barcos y usando Chuffed se demoró lo siguiente:

[illegible]

Mientras que con una entrada de 2 muelles, 12 barcos y usando Chuffed se demoró lo siguiente:

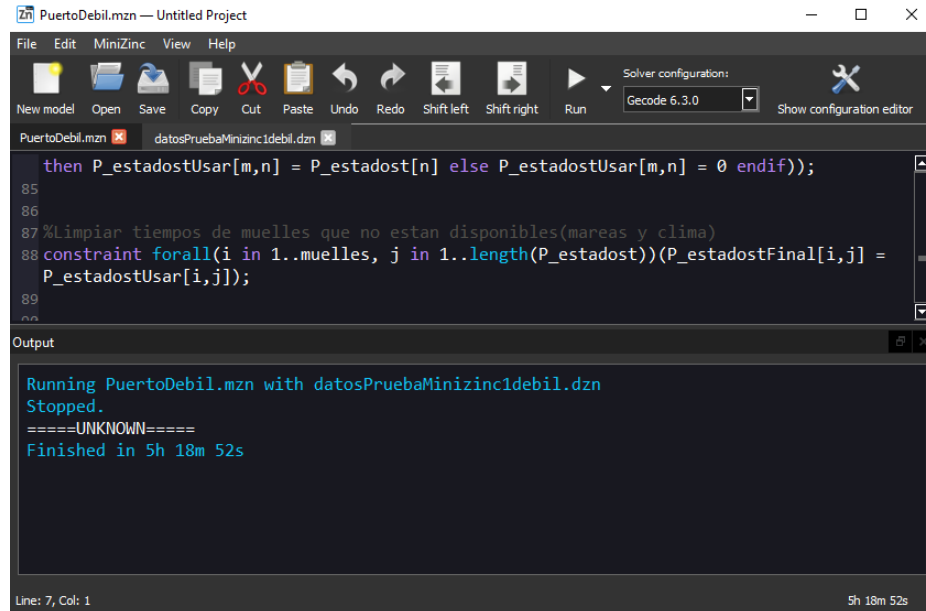
[illegible]

Por lo que se puede observar que, a pesar de tener una respuesta en un tiempo moderado bajo estos terminos, se debe tener cuidado con la entrada de este modelo porque a una mayor dificultad podra tener una respuesta va muy de-



morada.

Mientras que si se usa Gecode con la entrada de 2 muelles y 4 barcos tendremos que con 5 horas de ejecución aun no da una respuesta:



The screenshot shows the MiniZinc IDE interface. The title bar reads "PuertoDebil.mzn — Untitled Project". The menu bar includes "File", "Edit", "MiniZinc", "View", and "Help". The toolbar contains icons for "New model", "Open", "Save", "Copy", "Cut", "Paste", "Undo", "Redo", "Shift left", "Shift right", "Run", and a "Show configuration editor" button. The "Solver configuration" dropdown is set to "Gecode 6.3.0". The editor window shows the following code:

```
then P_estadostUsar[m,n] = P_estadost[n] else P_estadostUsar[m,n] = 0 endif));
85
86
87 %Limpiar tiempos de muelles que no estan disponibles(mareas y clima)
88 constraint forall(i in 1..muelles, j in 1..length(P_estadost))(P_estadostFinal[i,j] =
    P_estadostUsar[i,j]);
89
90
```

The "Output" window at the bottom displays the following text:

```
Running PuertoDebil.mzn with datosPruebaMinizinc1debil.dzn
Stopped.
=====UNKNOWN=====
Finished in 5h 18m 52s
```

The status bar at the bottom indicates "Line: 7, Col: 1" and "5h 18m 52s".

Y por último si se usa la entrada de 2 muelles y 12 barcos la respuesta se demorará mucho más que la anterior vista. Por lo que, para este modelo comparando estos dos solvers resulta mucho más eficiente el uso de Chuffed. Esto es debido a que Chuffed al ser un solver "lazy clause generation solver" para este modelo es capaz de decidir rapidamente si tomando un camino va a fallar; por lo que poda estos caminos, evita trabajo redundante y reduce considerablemente el tamaño total del árbol. [1]

### 3.3. Conclusiones planteamiento 2.

Escoger un solver es una parte bastante vital a la hora de solucionar problemas con el paradigma de restricciones. Pues de no elegir este correctamente, se puede llegar a ni siquiera tener respuesta para las entradas más "simples", como se logra observar en la comparación hecha anteriormente. Por lo que, se puede decir que en el modelo desarrollado en este segundo planteamiento se genera un árbol considerablemente grande porque su metodología misma es demasiado exhaustiva y por esto se ve beneficiado del uso del solver Chuffed y su estrategia. Está generación tan alta de ramas es debido a las restricciones que se tienen para manejar todo el tema del tiempo utilizado en el muelle y que el tiempo de un barco en todo el puerto sea continuo, pues estas restricciones tienen una complejidad considerablemente alta porque restringen cada elemento del arreglo con cada uno de los demas elementos en el arreglo.

## 4. Conclusiones generales.

Finalmente, al analizar los dos planteamientos desarrollados en este documento, se puede concluir que el *Planteamiento* 1 representa una solución mucho más rápida debido a que su metodología se basa en utilizar ciertas propiedades que se encuentran en el problema mismo, mientras que el *Planteamiento* 2 es una solución mucho más exhaustiva ya que está compuesta de múltiples restricciones que condicionan a cada uno de los elementos de varios arreglos, haciendo que la generación de ramas sea muchísimo mayor y al sucederle esto se ve beneficiado de solvers como Chuffed que su estrategia es capaz de podar y reducir rápidamente el tamaño de un árbol.

## Referencias

- [1] chuffed/chuffed, GitHub, 2021. [Online]. Available: <https://github.com/chuffed/chuffed>. [Accessed: 11- Jun- 2021]