MEMORIA DE LA PRÁCTICA

DE PLANIFICACIÓN

David Mingueza

Juanse Brito

Pere Vergés

**ÍNDICE**

[**1. Problema**](#_3jo4n9wh096f) **2**

[**2. Dominio**](#_zg8b2sbkuyc2) **3**

[2.1. Variables](#_qdjrvxt1p2o3) 3

[2.2. Funciones](#_atyjl147ubs8) 3

[2.3. Predicados](#_d8r832gfmfya) 3

[2.4. Acciones](#_7k2ng4k04g0a) 4

[**3. Modelización de los problemas**](#_j0yu7u8vxec1) **5**

[3.1. Nivel Básico](#_5e0va9csf2jw) 5

[3.2. Extensión 1](#_x0rzwvftymax) 6

[3.3. Extensión 2](#_z3bi2h88x9e5) 7

[3.4. Extensión 3](#_av4x2umf2giz) 9

[**4. Desarrollo de los problemas**](#_fze13svbcc32) **10**

[**5. Juegos de prueba**](#_qcovl9lhmu5v) **10**

[5.1. Nivel Básico](#_qi4gqeqda9be) 10

[5.2. Extensión 1](#_thlcwgjnpvxg) 10

5.3. Extensión 2 11

5.3.1 Versión básica 13

5.3.2 Versión optimización 17

5.4 Extensión 3 21

5.4.1 Versión maximizar prioridades 21

5.4.2 Versión maximizar prioridades y minimizar combustible 24

**6. Nota extra**

6.1 Juego de prueba 26

6.2 Experimiento tiempo de resolución 31

# **1. Problema**

El problema consiste en un conjunto de rovers situados en diferentes bases dentro de Marte y un conjunto de peticiones (de suministros o de personal) que deben ser servidas. El número de peticiones es igual o mayor que los recursos disponibles.

Existe un número de rovers, un número de bases, un número de recursos en cada base y un número de peticiones. Los rovers pueden cargar tantas personas y suministros como quieran.

La extensión 1 del problema limita a un suministro o dos personas el máximo de carga que un rover puede transportar.

La extensión 2 limita el número de viajes que un rover puede realizar en función de su combustible. También hay una segunda versión de esta extensión en la que se minimiza el combustible total gastado.

En la extensión 3 cada petición tiene un nivel de prioridad y se intenta maximizar la prioridad de las peticiones servidas.

Cada extensión incluye todas las anteriores.

# **2. Dominio**

## **2.1. Variables**

Tenemos dos variables distintas:

* **Base**: Indicar que el objeto se trata de una base, ya que hemos decidido distinguirlo en los predicados.
* **Rover**: Indica que el objeto se trata de un rover.

Las personas y los suministros son genéricos, no realizaremos operaciones sobre ninguno en concreto.

## **2.2. Funciones**

* **numeroDePersonas** ?R - rover, es el número de personas que hay en el rover R.

## **2.3. Predicados**

Estos son todos los predicados:

* **suministroEnBase** ?S - suministro ?B - base. Indica que el suministro S se encuentra en la base B.
* **personaEnBase** ?P - persona ?B - base. Indica que la persona P se encuentra en la base B.
* **roverEnBase** ?R - rover ?B - base. Indica que el rover R está situado en la base B.
* **personaEnRover** ?P - persona ?R - rover. Indica que la persona P está siendo transportada por el rover R.
* **suministroEnRover** ?S - suministro ?R - rover. Indica que el suministro S está siendo transportado por el rover R.
* **esAsentamineto** ?B - base. Indica que la base B es un asentamiento.
* **esAlmacen** ?B - base. Indica que la base B es un almacén.
* **roverCargaSuministro** ?R - rover, el rover R está cargando un suministro

## **2.4. Acciones**

En el caso base hemos decidido crear cinco acciones:

* **cargarPersona** - Carga una persona en un rover si está en una asentamiento, si la persona está en la misma base que el rover y si esta persona no se encuentra ya en un rover.
* **cargarSuministro** -Carga un suministro en un rover si está en un almacén, si el suministro está en la misma base que el rover y si el suministro no se encuentra ya en un rover.
* **descargarPersona** -Descarga de un rover una persona en un asentamiento.
* **descargaSuministro** -Descarga de un rover un suministro en un almacén.
* **moverRoverDeBase** - Mueve un rover de una base a otra, en el caso inicial hemos planeado que el rover pueda moverse de una base a cualquier otra, es decir, que todas las bases están conectadas entre ellas.

# **3. Modelización de los problemas**

En este apartado se describen los diferentes elementos que aparecen en cada extensión del problema, es decir, los objetos, estado inicial y el estado final que intervienen. Básicamente en cada extensión se han ido añadiendo funcionalidades a las extensiones anteriores, exceptuando a partir de la extensión 2 que se produce un cambio también de variables.

## **3.1. Nivel Básico**

**Objetos:**

* **Bases**: Son las bases que decidimos que habrá en el problema (b1,b2,b3…)
* **Suministro**: Son todos los suministros disponibles (s1,s2,s3...)
* **Persona**: Son las personas que habrá en el problema (p1,p2,p3…)
* **Rover**: Son los rovers que habrá en el problema (r1,r2,r3…)

**Estado inicial:**

* Primero definimos qué suministro está en qué base con el predicado ***suministroEnBase***
* Después definimos qué persona está en cada base con el predicado ***personaEnBase***y qué rover está en cada base con el predicado ***roverEnBase.***
* Por último definimos qué base es un almacén y qué base es un asentamiento con los predicados ***esAlmacen*** y ***esAsentamineto***

**Estado final:**

* El estado final es la serie de predicados de ***personaEnBase y suministroEnBase*** donde se especifica qué suministro o persona se quiere que llegue a cada base.

La solución que hemos propuesto es tener una serie de Bases (que definimos el tipo de la base con un predicado (que se podría hacer con un tipo)), una serie de suministros y personas (que podríamos tratarlo como Bases, es decir, hacer un solo tipo y luego distinguirlo con predicados) y finalmente una serie de rovers. El estado inicial básicamente es definir en qué base se encuentra cada rover, suministro y persona, y a partir de ahí crear el estado final, que es donde tiene que llegar cada suministro o persona. Para poder hacerlo tenemos las operaciones de cargar y descargar una persona o un suministro y mover un rover entre dos bases.

## **3.2. Extensión 1**

En esta extensión se debe tener en cuenta que como mucho pueden haber 2 personas o un suministro en un rover. Para ello hemos añadido una función que dice el número de personas que hay en un rover.

**Objetos:**

Los objetos son los mismos que en el nivel básico:

* **Bases**: Son las bases que decidimos que habrá en el problema (b1,b2,b3…)
* **Suministro**: Son todos los suministros disponibles (s1,s2,s3...)
* **Persona**: Son las personas que habrá en el problema (p1,p2,p3…)
* **Rover**: Son los rovers que habrá en el problema (r1,r2,r3…)

**Estado inicial:**

El estado inicial es prácticamente el mismo pero añadiendo el uso de la función ***numeroDePersonas***:

* Por cada rover le asignamos el ***numeroDePersonas*** que tiene en el estado inicial, que siempre será cero.
* Primero definimos qué suministro está en qué base con el predicado ***suministroEnBase.***
* Después definimos qué persona está en cada base con el predicado ***personaEnBase***y qué rover está en cada base con el predicado ***roverEnBase.***
* Por último definimos qué base es un almacén y qué base es un asentamiento con los predicados ***esAlmacen*** y ***esAsentamineto.***

**Estado final:**

El estado final es exactamente el mismo que en el de nivel básico:

* El estado final es la serie de predicados de ***personaEnBase y suministroEnBase*** donde se especifica qué suministro o persona se quiere que llegue a cada base.

Esta extensión es prácticamente como la anterior, pero con el añadido que en un rover como mucho puede haber o dos personas o un suministro, entonces la definición del estado inicial es la misma y le añadimos la función de que en un rover hay un número de personas (que al principio siempre es 0). Las acciones son las mismas pero tenemos en ellas un predicado que indica si un rover carga suministros o no, con esto podemos tratar la restricción de que no se sobrepase el número de personas y suministros en un rover.

## **3.3. Extensión 2**

La primera versión de esta extensión simplemente limita el número de viajes en función del combustible. El combustible de cada rover lo definimos en la inicialización del problema, y lo iremos decrementando con cada viaje que haga el rover.

No obstante, ahora se produce un cambio importante en el dominio. Este es debido a que al principio entendimos que las peticiones eran de una persona o suministro en concreto cuando debían ser generales.

Los cambios serán los siguientes:

* Variables: Eliminamos suministro y persona, ya que ahora lo trataremos como cantidades.
* Funciones:
  + **suministrosEnBase** ?B - base. Devuelve la cantidad de suministros para repartir que tiene la base B.
  + **personasEnBase** ?AS - base. Devuelve la cantidad de personal especializado para repartir que tiene la base AS.
  + **suministrosEnCirculacion.** Devuelve la cantidad total de suministros que estan en rovers esperando a ser entregados.
  + **personasEnCirculacion**. Devuelve la cantidad total de personal que esta en rovers esperando a ser entregados.
  + **peticionPersonaBase** ?B - base. Devuelve el número de peticiones de personal (cada petición es de una unidad) de la base B.
  + **peticionSuministroBase** ?B - base- Devuelve el número de peticiones de suministros de la base B.
  + **combustible** ?R - rover. Devuelve el combustible del rover R.
* Predicados:
  + **conexion** ?B1 - base, ?B2 - base. Un rover puede ir de la base B1 a la B2 porque hay una camino que las conecta.

Los demás predicados han sido eliminados, exceptuando roverEnBase, esAsentamiento, esAlmacen y roverCargaSuministro.

* Acciones:
  + **cargarPersona**. Ahora se comprueba en la precondicón que haya alguna persona para cargar en la base y el efecto es que se decrementa en uno las personas en la base y se incrementa el número de personas en circulación.
  + **cargarSuministro**. Idénticamente igual que cargarPersona pero con suministros.
  + **descargarPersona**. En la precondición se comprueba que la base donde está el rover haya hecho alguna petición de personal o le queden aún peticiones por satisfacer. El efecto disminuye en uno el número de peticiones de la base y también el personal en circulación.

En ningún caso con este predicado queremos que se aumente la cantidad de suministros o personal de la base ya que los predicados que usamos al principio son para objetos que han de ser entregados.

* + **descargarSuministro**. Idénticamente igual que descargarPersona.
  + **moverRoverDeBase**. Se comprobará en la precondición que el combustible del rover sea superior a 1, y en caso de realizarse, se decrementará en 1 el combustible. Además, evidentemente, de comprobar que las bases están conectadas.

Añadimos también que antes los suministros viajaban de almacén a almacén, ahora viajan de almacén a asentamiento como dice el enunciado.

Al no definir conexiones entre las bases, tenemos que todas están conectadas con coste 1, que hace el problema trivial por un lado, pero por otra parte, a la hora de optimizar provocará que haya de probar una gran cantidad de combinaciones. Por lo tanto ahora se añade un predicado conexión entre dos bases.

**Objetos:**

Los objetos se han reducido a los siguientes:

* **Bases**: Son las bases que decidimos que habrá en el problema (b1,b2,b3…)
* **Rovers**: Son los rovers que habrá en el problema (r1,r2,r3…)

**Estado inicial:**

Primero, se define la información relevante a las bases, con el predicado ***conexion***, definiendo qué bases están conectadas y qué base es almacén y asentamiento con ***esAlmacen*** y ***esAsentamiento***. También se indica, si es un almacén, la cantidad de suministros para ser transportados que contiene con la función ***suministrosEnBase*** y, si es un asentamiento, la cantidad de suministros que contiene con ***personasEnBase***.

A continuación, respecto a los rovers, indicamos como en la extensión anterior en que base está y el número de personas que está transportando. Pero se incluye el combustible inicial del que dispone con la función combustible.

También definimos la cantidad de peticiones de cada base si tienen alguna con la función ***peticionSuministroBase*** y ***peticionPersonaBase***.

Por último, establecemos a 0 el número de personas y suministros en circulación con ***personasEnCirculacion*** y ***suministrosEnCirculacion***.

**Estado final:**

Los estados finales de las otras extensiones eran realmente ineficientes porque tenían que evaluar muchas condiciones. Ahora con cuatro comprobamos todo:

* Un forall para todas las bases indicando que si es almacén implica que sus ***suministrosEnBase*** sean 0, que significa que se han cargado todos en algún rover, si tenían alguno. Por otro lado, también comprueba que si es asentamiento implica que sus ***personasEnBase*** sean 0.
* Condición que indica que las ***personasEnCirculacion*** son 0.
* Condición que indica que los ***suministrosEnCirculacion*** son 0.

Con estos predicados queremos expresar lo siguiente: el plan acaba cuando el suministro y personal de las bases ha sido cargado en algún rover y ya no está en circulación porque todo ha conseguido llegar a algún destino.

Esta meta es correcta siempre y cuando haya más peticiones que suministros y personal del que se puede servir, tal y como dice el enunciado.

La segunda versión de la extensión tiene el añadido de que se debe minimizar el gasto de combustible total. Para ello se añade la siguiente función:

* Funciones:
  + **total-cost**. Es el cantidad total de combustible gastada por todos los rovers.
* Acciones:
  + **moverRoverDeBase.** Ahora en el efecto se incluye que cada vez que se mueva un rover se incremente ***total-cost***.

En el estado inicial se inicializa a 0 ***total-cost***, y ahora se añade la siguiente línea al problema después de *goal*: *(:metric minimize (total-cost))*

## **3.4. Extensión 3**

En esta extensión se establecen prioridades a las peticiones, de forma que se busca maximizar la prioridad de las peticiones servidas.

Para modelar esta información hemos decidido que aplicamos la prioridad a un conjunto de peticiones y no a peticiones individuales. De forma que, si una base ha hecho 9 peticiones de suministros, este conjunto de peticiones tendrá una prioridad. Lo mismo pasa con las solicitudes de personal.

Los objetos y el estado final siguen igual que en la extensión 2.

En el estado inicial hemos añadido una función ***penalizacion-total***. En los efectos de los operadores de descargar personas y suministros, ahora establecemos unas condiciones tal que si las peticiones tienen prioridad 3 no se suma nada a la función penalización. Si tiene prioridad 2 se suma 5, y si tiene prioridad 1 se suma 10.

Al final, en el fichero del problema se pide minimizar esta función. Con esto obtenemos que contra menos prioridad tenga una petición le llegarán menos suministros y personal ante peticiones con más prioridad.

Para la segunda versión de esta extensión, cambiamos lo que definimos en el *metric* de forma que ahora se puede combinar prioridades y combustible total dándoles un peso a cada uno. Esto lo hacemos con la siguiente línea de código:

*(:metric minimize (+ (\* 1 (penalizacion-total)) (\* 10 (total-cost))))*

Donde damos un peso multiplicando por un factor y luego minimizamos la suma de ambos.

# **4. Desarrollo de los problemas**

Para el desarrollo de los problemas que había en esta práctica lo que hemos decidido es empezar por el primero, que sería como el “caso base” de la práctica y a partir de este añadir pequeñas modificaciones a cada extensión que hemos hecho del problema para que cumpla sus funcionalidades. Hemos escogido hacerlo de este modo puesto que el enunciado iba muy enfocado a este desarrollo, a ir aumentando poco a poco la dificultad de cada extensión.

Esta era nuestra idea inicial, pero visto el problema que hemos detectado a partir de la extensión 2 hemos decidido hacer un cambio más significativo para alcanzar mejor el objetivo que expresa el enunciado.

# **5. Juegos de prueba**

## **5.1. Nivel Básico**

Juego de pruebas muy básico en el que tenemos un almacén b3 y una asentamiento b4, que necesitan que le lleguen los suministros (s1,s2,s3) y personas (p1,p2,p3) y solo tenemos un rover en la base b1. Entonces este juego de pruebas es para ver como hace el rover para el solo hacer todas las tareas (este juego es para ver la diferencia luego con la extensión 1).

(:objects

b1 b2 b3 b4 - base

s1 s2 s3 - suministro

p1 p2 p3 - persona

r1 - rover

)

(:init

(roverEnBase r1 b1)

(suministroEnBase s1 b1)

(suministroEnBase s2 b1)

(suministroEnBase s3 b1)

(personaEnBase p1 b2)

(personaEnBase p2 b2)

(personaEnBase p3 b2)

(esAlmacen b1)

(esAlmacen b3)

(esAsentamineto b2)

(esAsentamineto b4)

)

(:goal

(and

(personaEnBase p1 b4)

(personaEnBase p2 b4)

(personaEnBase p3 b4)

(suministroEnBase s1 b3)

(suministroEnBase s2 b3)

(suministroEnBase s3 b3)

)

)

**Output:**

0: CARGARSUMINISTRO S1 R1 B1

1: CARGARSUMINISTRO S2 R1 B1

2: CARGARSUMINISTRO S3 R1 B1

3: MOVERROVERDEBASE R1 B1 B2

4: CARGARPERSONA P1 R1 B2

5: CARGARPERSONA P2 R1 B2

6: CARGARPERSONA P3 R1 B2

7: MOVERROVERDEBASE R1 B2 B4

8: DECARGARPERSONA P1 R1 B4

9: DECARGARPERSONA P2 R1 B4

10: DECARGARPERSONA P3 R1 B4

11: MOVERROVERDEBASE R1 B4 B3

12: DECARGARSUMINISTRO S1 R1 B3

13: DECARGARSUMINISTRO S2 R1 B3

14: DECARGARSUMINISTRO S3 R1 B3

Entonces vemos como el rover carga los suministros de B1, se mueve a B2, carga las personas, se mueve a B4, descarga las personas, y finalmente se mueve a B3 y descarga los suministros. Era el resultado esperado puesto que puede cargar tantos suministros y personal como quiera.

## **5.2. Extensión 1**

En este juego de pruebas queremos hacer la comparación con el juego de pruebas del nivel básico, y ver la diferencia del comportamiento de un solo rover entre no tener restricción de carga y con tener restricción de dos personas o de una carga de suministro.

(:objects

b1 b2 b3 b4 - base

s1 s2 s3 - suministro

p1 p2 p3 - persona

r1 - rover

)

(:init

(roverEnBase r1 b1)

(= (numeroDePersonas r1) 0)

(suministroEnBase s1 b1)

(suministroEnBase s2 b1)

(suministroEnBase s3 b1)

(personaEnBase p1 b2)

(personaEnBase p2 b2)

(personaEnBase p3 b2)

(esAlmacen b1)

(esAlmacen b3)

(esAsentamiento b2)

(esAsentamiento b4)

)

(:goal

(and

(personaEnBase p1 b4)

(personaEnBase p2 b4)

(personaEnBase p3 b4)

(suministroEnBase s1 b3)

(suministroEnBase s2 b3)

(suministroEnBase s3 b3)

)

)

**Output:**

0: CARGARSUMINISTRO S1 R1 B1

1: MOVERROVERDEBASE R1 B1 B2

2: MOVERROVERDEBASE R1 B2 B3

3: DESCARGARSUMINISTRO S1 R1 B3

4: MOVERROVERDEBASE R1 B3 B1

5: CARGARSUMINISTRO S2 R1 B1

6: MOVERROVERDEBASE R1 B1 B2

7: MOVERROVERDEBASE R1 B2 B3

8: DESCARGARSUMINISTRO S2 R1 B3

9: MOVERROVERDEBASE R1 B3 B1

10: CARGARSUMINISTRO S3 R1 B1

11: MOVERROVERDEBASE R1 B1 B3

12: DESCARGARSUMINISTRO S3 R1 B3

13: MOVERROVERDEBASE R1 B3 B2

14: CARGARPERSONA P1 R1 B2

15: CARGARPERSONA P2 R1 B2

16: MOVERROVERDEBASE R1 B2 B4

17: DESCARGARPERSONA P1 R1 B4

18: MOVERROVERDEBASE R1 B4 B2

19: CARGARPERSONA P3 R1 B2

20: MOVERROVERDEBASE R1 B2 B4

21: DESCARGARPERSONA P2 R1 B4

22: DESCARGARPERSONA P3 R1 B4

En este caso podemos ver como el rover cada vez que carga un suministro se mueve a la base que le corresponde para descargar la carga, y que como mucho carga dos personas, por lo tanto vemos que cumple las restricciones propuestas. Además, podemos observar una gran diferencia de comportamiento con el nivel básico, puesto que en este caso el rover hace muchos más movimientos para no sobrepasar la carga establecida. Pero podemos observar un comportamiento extraño, alguna vez hace dos movimientos, y en esta extensión se puede llegar a cualquier base en un solo movimiento, hemos hecho algunas pruebas y creemos que es debido a que no están definidas las conexiones.

## **5.3.Extensión 2**

## **5.3.1 Versión básica**

Fichero del problema:

(:domain marte)

(:objects

b1 b2 b3 - base

r1 r2 - rover

)

(:init

(conexion b1 b2)

(conexion b2 b3)

(conexion b3 b1)

(esAlmacen b1)

(esAsentamiento b2)

(esAsentamiento b3)

(= (suministrosEnBase b1) 1)

(= (personasEnBase b2) 20)

(= (personasEnBase b3) 0)

(roverEnBase r1 b1)

(= (numeroDePersonas r1) 0)

(= (combustible r1) 60)

(roverEnBase r2 b2)

(= (numeroDePersonas r2) 0)

(= (combustible r2) 60)

(= (peticionSuministroBase b3) 30) ;numero de peticiones de la base, respecto a suministros

(= (peticionPersonaBase b3) 30)

(= (personasEnCirculacion) 0) ;estan en algun rover esperando llegar a su destino

(= (suministrosEnCirculacion) 0)

)

(:goal

(and

(forall (?b - base) (or (not(esAlmacen ?b)) (= (suministrosEnBase ?b) 0))) ; almacen implica suministrosEnBase=0, traducido logicamente

(forall (?b - base) (or (not(esAsentamiento ?b)) (= (personasEnBase ?b) 0))) ; asentamiento implica personasEnBase=0

(= (personasEnCirculacion) 0)

(= (suministrosEnCirculacion) 0)

)

)

**Output:**

0: MOVERROVERDEBASE R1 B1 B2

1: CARGARPERSONA R1 B2

2: MOVERROVERDEBASE R1 B2 B3

3: DESCARGARPERSONA R1 B3

4: MOVERROVERDEBASE R1 B3 B1

5: MOVERROVERDEBASE R1 B1 B2

6: CARGARPERSONA R1 B2

7: MOVERROVERDEBASE R1 B2 B3

8: DESCARGARPERSONA R1 B3

9: MOVERROVERDEBASE R1 B3 B1

10: MOVERROVERDEBASE R1 B1 B2

11: CARGARPERSONA R1 B2

12: MOVERROVERDEBASE R1 B2 B3

13: DESCARGARPERSONA R1 B3

14: MOVERROVERDEBASE R1 B3 B1

15: MOVERROVERDEBASE R1 B1 B2

16: CARGARPERSONA R1 B2

17: MOVERROVERDEBASE R1 B2 B3

18: DESCARGARPERSONA R1 B3

19: MOVERROVERDEBASE R1 B3 B1

20: MOVERROVERDEBASE R1 B1 B2

21: CARGARPERSONA R1 B2

22: MOVERROVERDEBASE R1 B2 B3

23: DESCARGARPERSONA R1 B3

24: MOVERROVERDEBASE R1 B3 B1

25: MOVERROVERDEBASE R1 B1 B2

26: CARGARPERSONA R1 B2

27: MOVERROVERDEBASE R1 B2 B3

28: DESCARGARPERSONA R1 B3

29: MOVERROVERDEBASE R1 B3 B1

30: MOVERROVERDEBASE R1 B1 B2

31: CARGARPERSONA R1 B2

32: MOVERROVERDEBASE R1 B2 B3

33: DESCARGARPERSONA R1 B3

34: MOVERROVERDEBASE R1 B3 B1

35: MOVERROVERDEBASE R1 B1 B2

36: CARGARPERSONA R1 B2

37: MOVERROVERDEBASE R1 B2 B3

38: DESCARGARPERSONA R1 B3

39: MOVERROVERDEBASE R1 B3 B1

40: MOVERROVERDEBASE R1 B1 B2

41: CARGARPERSONA R1 B2

42: MOVERROVERDEBASE R1 B2 B3

43: DESCARGARPERSONA R1 B3

44: MOVERROVERDEBASE R1 B3 B1

45: MOVERROVERDEBASE R1 B1 B2

46: CARGARPERSONA R1 B2

47: MOVERROVERDEBASE R1 B2 B3

48: DESCARGARPERSONA R1 B3

49: MOVERROVERDEBASE R1 B3 B1

50: MOVERROVERDEBASE R1 B1 B2

51: CARGARPERSONA R1 B2

52: MOVERROVERDEBASE R1 B2 B3

53: DESCARGARPERSONA R1 B3

54: MOVERROVERDEBASE R1 B3 B1

55: MOVERROVERDEBASE R1 B1 B2

56: CARGARPERSONA R1 B2

57: MOVERROVERDEBASE R1 B2 B3

58: DESCARGARPERSONA R1 B3

59: MOVERROVERDEBASE R1 B3 B1

60: MOVERROVERDEBASE R1 B1 B2

61: CARGARPERSONA R1 B2

62: MOVERROVERDEBASE R1 B2 B3

63: DESCARGARPERSONA R1 B3

64: MOVERROVERDEBASE R1 B3 B1

65: MOVERROVERDEBASE R1 B1 B2

66: CARGARPERSONA R1 B2

67: MOVERROVERDEBASE R1 B2 B3

68: DESCARGARPERSONA R1 B3

69: MOVERROVERDEBASE R1 B3 B1

70: MOVERROVERDEBASE R1 B1 B2

71: CARGARPERSONA R1 B2

72: MOVERROVERDEBASE R1 B2 B3

73: DESCARGARPERSONA R1 B3

74: MOVERROVERDEBASE R1 B3 B1

75: MOVERROVERDEBASE R1 B1 B2

76: CARGARPERSONA R1 B2

77: MOVERROVERDEBASE R1 B2 B3

78: DESCARGARPERSONA R1 B3

79: MOVERROVERDEBASE R1 B3 B1

80: MOVERROVERDEBASE R1 B1 B2

81: CARGARPERSONA R2 B2

82: MOVERROVERDEBASE R2 B2 B3

83: DESCARGARPERSONA R2 B3

84: MOVERROVERDEBASE R2 B3 B1

85: CARGARSUMINISTRO R2 B1

86: MOVERROVERDEBASE R2 B1 B2

87: MOVERROVERDEBASE R2 B2 B3

88: DESCARGARSUMINISTRO R2 B3

89: CARGARPERSONA R1 B2

90: MOVERROVERDEBASE R1 B2 B3

91: MOVERROVERDEBASE R2 B3 B1

92: MOVERROVERDEBASE R2 B1 B2

93: DESCARGARPERSONA R1 B3

94: CARGARPERSONA R2 B2

95: CARGARPERSONA R2 B2

96: MOVERROVERDEBASE R2 B2 B3

97: DESCARGARPERSONA R2 B3

98: DESCARGARPERSONA R2 B3

El objetivo de este juego de pruebas es observar como el planificador resuelve el problema con una cantidad de personas más elevada manteniendo las restricciones. Como podemos ver se respetan adecuadamente las restricciones contempladas en las anteriores extensiones y que los rovers se mueven correctamente por las conexiones definidas.

Hemos decidido centrarnos en cómo transporta las personas para ver si en todo momento podía hacer un plan cumpliendo las restricciones, como por ejemplo, que la carga máxima de personas en un rover es 2.

## **5.3.2 Versión optimización**

Fichero de problema:

(define (problem roversEnMarte)

(:domain marte)

(:objects

b1 b2 b3 - base

r1 r2 - rover

)

(:init

(conexion b1 b2)

(conexion b2 b3)

(conexion b3 b1)

(esAlmacen b1)

(esAsentamiento b2)

(esAsentamiento b3)

(= (suministrosEnBase b1) 1)

(= (personasEnBase b2) 20)

(= (personasEnBase b3) 0)

(roverEnBase r1 b1)

(= (numeroDePersonas r1) 0)

(= (combustible r1) 60)

(roverEnBase r2 b2)

(= (numeroDePersonas r2) 0)

(= (combustible r2) 60)

(= (peticionSuministroBase b3) 30) ;numero de peticiones de la base, respecto a suministros

(= (peticionPersonaBase b3) 30)

(= (personasEnCirculacion) 0) ;estan en algun rover esperando llegar a su destino

(= (suministrosEnCirculacion) 0)

(= (total-cost) 0)

)

(:goal

(and

(forall (?b - base) (or (not(esAlmacen ?b)) (= (suministrosEnBase ?b) 0))) ; almacen implica suministrosEnBase=0, traducido logicamente

(forall (?b - base) (or (not(esAsentamiento ?b)) (= (personasEnBase ?b) 0))) ; asentamiento implica personasEnBase=0

(= (personasEnCirculacion) 0)

(= (suministrosEnCirculacion) 0)

)

)

(:metric minimize (total-cost))

)

**Output:**

0: CARGARPERSONA R2 B2

1: CARGARPERSONA R2 B2

2: MOVERROVERDEBASE R2 B2 B3

3: DESCARGARPERSONA R2 B3

4: DESCARGARPERSONA R2 B3

5: MOVERROVERDEBASE R2 B3 B1

6: MOVERROVERDEBASE R2 B1 B2

7: CARGARPERSONA R2 B2

8: CARGARPERSONA R2 B2

9: MOVERROVERDEBASE R2 B2 B3

10: DESCARGARPERSONA R2 B3

11: DESCARGARPERSONA R2 B3

12: MOVERROVERDEBASE R2 B3 B1

13: MOVERROVERDEBASE R2 B1 B2

14: CARGARPERSONA R2 B2

15: CARGARPERSONA R2 B2

16: MOVERROVERDEBASE R2 B2 B3

17: DESCARGARPERSONA R2 B3

18: DESCARGARPERSONA R2 B3

19: MOVERROVERDEBASE R2 B3 B1

20: MOVERROVERDEBASE R2 B1 B2

21: CARGARPERSONA R2 B2

22: CARGARPERSONA R2 B2

23: MOVERROVERDEBASE R2 B2 B3

24: DESCARGARPERSONA R2 B3

25: DESCARGARPERSONA R2 B3

26: MOVERROVERDEBASE R2 B3 B1

27: MOVERROVERDEBASE R2 B1 B2

28: CARGARPERSONA R2 B2

29: CARGARPERSONA R2 B2

30: MOVERROVERDEBASE R2 B2 B3

31: DESCARGARPERSONA R2 B3

32: DESCARGARPERSONA R2 B3

33: MOVERROVERDEBASE R2 B3 B1

34: MOVERROVERDEBASE R2 B1 B2

35: CARGARPERSONA R2 B2

36: CARGARPERSONA R2 B2

37: MOVERROVERDEBASE R2 B2 B3

38: DESCARGARPERSONA R2 B3

39: DESCARGARPERSONA R2 B3

40: MOVERROVERDEBASE R2 B3 B1

41: MOVERROVERDEBASE R2 B1 B2

42: CARGARPERSONA R2 B2

43: CARGARPERSONA R2 B2

44: MOVERROVERDEBASE R2 B2 B3

45: DESCARGARPERSONA R2 B3

46: DESCARGARPERSONA R2 B3

47: MOVERROVERDEBASE R2 B3 B1

48: MOVERROVERDEBASE R2 B1 B2

49: CARGARPERSONA R2 B2

50: CARGARPERSONA R2 B2

51: MOVERROVERDEBASE R2 B2 B3

52: DESCARGARPERSONA R2 B3

53: MOVERROVERDEBASE R1 B1 B2

54: DESCARGARPERSONA R2 B3

55: MOVERROVERDEBASE R2 B3 B1

56: CARGARSUMINISTRO R2 B1

57: MOVERROVERDEBASE R2 B1 B2

58: MOVERROVERDEBASE R2 B2 B3

59: DESCARGARSUMINISTRO R2 B3

60: MOVERROVERDEBASE R2 B3 B1

61: MOVERROVERDEBASE R2 B1 B2

62: CARGARPERSONA R2 B2

63: MOVERROVERDEBASE R2 B2 B3

64: DESCARGARPERSONA R2 B3

65: CARGARPERSONA R1 B2

66: MOVERROVERDEBASE R2 B3 B1

67: MOVERROVERDEBASE R2 B1 B2

68: MOVERROVERDEBASE R1 B2 B3

69: CARGARPERSONA R2 B2

70: CARGARPERSONA R2 B2

71: MOVERROVERDEBASE R2 B2 B3

72: DESCARGARPERSONA R1 B3

73: DESCARGARPERSONA R2 B3

74: DESCARGARPERSONA R2 B3

El objetivo de este juego es observar cómo actúa la optimización. Para ello cogemos el mismo problema que el usado en la versión básica de esta extensión.

A primera vista, vemos que ha resuelto el problema en muchos menos pasos. Y si contamos las veces que usa la acción de mover, tenemos:

* Versión básica: **57** desplazamientos
* Versión optimización: **33** desplazamientos

El resultado es bastante exitoso ya que ha conseguido reducir más de la mitad de los desplazamientos totales. Añadimos también que tarda más tiempo en conseguir optimizar, esto se debe a que teniendo en cuenta que simplemente hay tres bases conectadas con un par de caminos de coste 1, el planificador tiene bastantes combinaciones que son parecidas y óptimas también y como las comprueba todas tarda un poco más en generar un plan.

Por último, en esta versión se siguen respetando las restricciones consideradas anteriormente ya que únicamente cambia del código la definición de un coste total.

## 

## **5.3.Extensión 3**

**5.3.1 Versión maximizar prioridades**

Fichero del problema:

(define (problem roversEnMarte)

(:domain marte)

(:objects

b1 b2 b3 b4 - base

r1 r2 - rover

)

(:init

(conexion b1 b2)

(conexion b2 b3)

(conexion b3 b4)

(conexion b4 b1)

(esAlmacen b1)

(esAsentamiento b2)

(esAsentamiento b3)

(esAsentamiento b4)

(= (suministrosEnBase b1) 5)

(= (personasEnBase b2) 4)

(= (personasEnBase b3) 0)

(= (personasEnBase b4) 0)

(roverEnBase r1 b1)

(= (numeroDePersonas r1) 0)

(= (combustible r1) 50)

(roverEnBase r2 b2)

(= (numeroDePersonas r2) 0)

(= (combustible r2) 50)

(= (peticionSuministroBase b3) 30)

(= (prioridadSuministroBase b3) 3)

(= (peticionPersonaBase b3) 30)

(= (prioridadPersonaBase b3) 3)

(= (peticionSuministroBase b4) 30)

(= (prioridadSuministroBase b4) 1)

(= (peticionPersonaBase b4) 30)

(= (prioridadPersonaBase b4) 1)

(= (personasEnCirculacion) 0) ;estan en algun rover esperando llegar a su destino

(= (suministrosEnCirculacion) 0)

(= (total-cost) 0)

(= (penalizacion-total) 0)

)

(:goal

(and

(forall (?b - base) (and (or (not(esAsentamiento ?b)) (= (personasEnBase ?b) 0)) (or (not(esAlmacen ?b)) (= (suministrosEnBase ?b) 0))))

(= (suministrosEnCirculacion) 0)

(= (personasEnCirculacion) 0)

)

)

(:metric minimize (penalizacion-total))

)

**Output:**

0: CARGARSUMINISTRO R1 B1

1: CARGARPERSONA R2 B2

2: MOVERROVERDEBASE R1 B1 B2

3: MOVERROVERDEBASE R1 B2 B3

4: MOVERROVERDEBASE R2 B2 B3

5: DESCARGARPERSONA R2 B3

6: MOVERROVERDEBASE R2 B3 B4

7: MOVERROVERDEBASE R2 B4 B1

8: MOVERROVERDEBASE R2 B1 B2

9: CARGARPERSONA R2 B2

10: MOVERROVERDEBASE R2 B2 B3

11: DESCARGARPERSONA R2 B3

12: MOVERROVERDEBASE R2 B3 B4

13: MOVERROVERDEBASE R2 B4 B1

14: MOVERROVERDEBASE R2 B1 B2

15: CARGARPERSONA R2 B2

16: MOVERROVERDEBASE R2 B2 B3

17: DESCARGARPERSONA R2 B3

18: MOVERROVERDEBASE R2 B3 B4

19: MOVERROVERDEBASE R2 B4 B1

20: MOVERROVERDEBASE R2 B1 B2

21: CARGARPERSONA R2 B2

22: MOVERROVERDEBASE R2 B2 B3

23: DESCARGARPERSONA R2 B3

24: MOVERROVERDEBASE R2 B3 B4

25: MOVERROVERDEBASE R2 B4 B1

26: MOVERROVERDEBASE R2 B1 B2

27: MOVERROVERDEBASE R2 B2 B3

28: MOVERROVERDEBASE R2 B3 B4

29: MOVERROVERDEBASE R2 B4 B1

30: MOVERROVERDEBASE R2 B1 B2

31: MOVERROVERDEBASE R2 B2 B3

32: MOVERROVERDEBASE R2 B3 B4

33: MOVERROVERDEBASE R2 B4 B1

34: MOVERROVERDEBASE R2 B1 B2

35: MOVERROVERDEBASE R2 B2 B3

36: MOVERROVERDEBASE R2 B3 B4

37: MOVERROVERDEBASE R2 B4 B1

38: MOVERROVERDEBASE R2 B1 B2

39: MOVERROVERDEBASE R2 B2 B3

40: MOVERROVERDEBASE R2 B3 B4

41: MOVERROVERDEBASE R2 B4 B1

42: MOVERROVERDEBASE R2 B1 B2

43: MOVERROVERDEBASE R2 B2 B3

44: MOVERROVERDEBASE R2 B3 B4

45: MOVERROVERDEBASE R2 B4 B1

46: MOVERROVERDEBASE R2 B1 B2

47: MOVERROVERDEBASE R2 B2 B3

48: MOVERROVERDEBASE R2 B3 B4

49: MOVERROVERDEBASE R2 B4 B1

50: MOVERROVERDEBASE R2 B1 B2

51: MOVERROVERDEBASE R2 B2 B3

52: MOVERROVERDEBASE R2 B3 B4

53: MOVERROVERDEBASE R2 B4 B1

54: MOVERROVERDEBASE R2 B1 B2

55: MOVERROVERDEBASE R2 B2 B3

56: MOVERROVERDEBASE R2 B3 B4

57: MOVERROVERDEBASE R2 B4 B1

58: MOVERROVERDEBASE R2 B1 B2

59: MOVERROVERDEBASE R2 B2 B3

60: MOVERROVERDEBASE R2 B3 B4

61: DESCARGARSUMINISTRO R1 B3

62: MOVERROVERDEBASE R1 B3 B4

63: MOVERROVERDEBASE R1 B4 B1

64: CARGARSUMINISTRO R1 B1

65: MOVERROVERDEBASE R1 B1 B2

66: MOVERROVERDEBASE R1 B2 B3

67: DESCARGARSUMINISTRO R1 B3

68: MOVERROVERDEBASE R1 B3 B4

69: MOVERROVERDEBASE R1 B4 B1

70: CARGARSUMINISTRO R1 B1

71: MOVERROVERDEBASE R1 B1 B2

72: MOVERROVERDEBASE R1 B2 B3

73: DESCARGARSUMINISTRO R1 B3

74: MOVERROVERDEBASE R1 B3 B4

75: MOVERROVERDEBASE R1 B4 B1

76: CARGARSUMINISTRO R1 B1

77: MOVERROVERDEBASE R1 B1 B2

78: MOVERROVERDEBASE R1 B2 B3

79: DESCARGARSUMINISTRO R1 B3

80: MOVERROVERDEBASE R1 B3 B4

81: MOVERROVERDEBASE R1 B4 B1

82: CARGARSUMINISTRO R1 B1

83: MOVERROVERDEBASE R1 B1 B2

84: MOVERROVERDEBASE R1 B2 B3

85: DESCARGARSUMINISTRO R1 B3

El objetivo de este juego de pruebas es ver si la base b3 se lleva todos los suministros y personal disponibles ya que sus peticiones tienen mucha más prioridad que las de la base b4.

Observamos que no se ha producido ninguna descarga en la base b4 y todo ha sido suministrado a b3 que es lo que esperábamos. Por otro lado, vemos que los rovers hacen muchos pasos innecesarios ya que no se tiene en cuenta el combustible total gastado.

**5.3.2 Versión maximizar prioridades y minimizar combustible**

Fichero del problema:

Cojemos el mismo fichero que en el juego de pruebas anterior pero cambiando la línea donde se define el metric por:

(:metric minimize (+ (\* 1 (penalizacion-total)) (\* 10 (total-cost))))

**Output:**

0: CARGARPERSONA R2 B2

1: CARGARPERSONA R2 B2

2: MOVERROVERDEBASE R2 B2 B3

3: DESCARGARPERSONA R2 B3

4: DESCARGARPERSONA R2 B3

5: MOVERROVERDEBASE R2 B3 B4

6: MOVERROVERDEBASE R2 B4 B1

7: CARGARSUMINISTRO R1 B1

8: MOVERROVERDEBASE R1 B1 B2

9: MOVERROVERDEBASE R1 B2 B3

10: DESCARGARSUMINISTRO R1 B3

11: MOVERROVERDEBASE R1 B3 B4

12: MOVERROVERDEBASE R1 B4 B1

13: MOVERROVERDEBASE R1 B1 B2

14: CARGARPERSONA R1 B2

15: CARGARPERSONA R1 B2

16: MOVERROVERDEBASE R1 B2 B3

17: DESCARGARPERSONA R1 B3

18: DESCARGARPERSONA R1 B3

19: MOVERROVERDEBASE R1 B3 B4

20: MOVERROVERDEBASE R1 B4 B1

21: CARGARSUMINISTRO R1 B1

22: MOVERROVERDEBASE R1 B1 B2

23: MOVERROVERDEBASE R1 B2 B3

24: DESCARGARSUMINISTRO R1 B3

25: MOVERROVERDEBASE R1 B3 B4

26: MOVERROVERDEBASE R1 B4 B1

27: CARGARSUMINISTRO R2 B1

28: MOVERROVERDEBASE R2 B1 B2

29: MOVERROVERDEBASE R2 B2 B3

30: CARGARSUMINISTRO R1 B1

31: MOVERROVERDEBASE R1 B1 B2

32: MOVERROVERDEBASE R1 B2 B3

33: DESCARGARSUMINISTRO R1 B3

34: MOVERROVERDEBASE R1 B3 B4

35: MOVERROVERDEBASE R1 B4 B1

36: CARGARSUMINISTRO R1 B1

37: MOVERROVERDEBASE R1 B1 B2

38: MOVERROVERDEBASE R1 B2 B3

39: DESCARGARSUMINISTRO R1 B3

40: DESCARGARSUMINISTRO R2 B3

En esta prueba cojemos cambiamos el metric asignándole bastante peso al coste total de combustible para ver si elimina pasos innecesarios.

Como esperábamos a reducido enormemente la cantidad de acciones de mover rover, de 65 a **23**. A más la base b4 sigue sin recibir ningún suministro ni persona.

# **6. Nota extra**

Incialmente elaboramos los juegos de pruebas a mano sin el uso del generador, ya que al ser un apartado opcional decidimos organizarnos y dejarlo para el final si disponíamos del tiempo necesario.

Finalmente lo hemos hecho y como demostración haremos un juego de pruebas para la versión básica de la extensión 2.

Aunque en el generador define las funciones que utilizamos en la extensión 3, lo usamos para hacer un juego de pruebas de una extensión anterior ya que es compatible.

## **6.1.Juego de prueba**

Problema generado con la opción -b 8 y -r 5:

(define (problem roversEnMarte)

(:domain marte)

(:objects b0 b1 b2 b3 b4 b5 b6 b7 -base

r0 r1 r2 r3 r4 -rover

)

(:init

(conexion b0 b1)

(conexion b1 b2)

(conexion b2 b3)

(conexion b3 b4)

(conexion b4 b5)

(conexion b5 b6)

(conexion b6 b7)

(conexion b7 b0)

(esAlmacen b0)

(= (suministrosEnBase b0) 0)

(esAlmacen b1)

(= (suministrosEnBase b1) 4)

(esAsentamiento b2)

(= (personasEnBase b2) 8)

(esAsentamiento b3)

(= (personasEnBase b3) 5)

(esAsentamiento b4)

(= (personasEnBase b4) 0)

(esAsentamiento b5)

(= (personasEnBase b5) 0)

(esAsentamiento b6)

(= (personasEnBase b6) 0)

(esAsentamiento b7)

(= (personasEnBase b7) 0)

(roverEnBase r0 b5)

(= (numeroDePersonas r0) 0)

(= (combustible r0) 78)

(roverEnBase r1 b0)

(= (numeroDePersonas r1) 0)

(= (combustible r1) 297)

(roverEnBase r2 b1)

(= (numeroDePersonas r2) 0)

(= (combustible r2) 87)

(roverEnBase r3 b6)

(= (numeroDePersonas r3) 0)

(= (combustible r3) 159)

(roverEnBase r4 b5)

(= (numeroDePersonas r4) 0)

(= (combustible r4) 113)

(= (peticionSuministroBase b4) 488)

(= (prioridadSuministroBase b4) 1)

(= (peticionPersonaBase b4) 155)

(= (prioridadPersonaBase b4) 1)

(= (peticionSuministroBase b5) 818)

(= (prioridadSuministroBase b5) 3)

(= (peticionPersonaBase b5) 53)

(= (prioridadPersonaBase b5) 3)

(= (peticionSuministroBase b6) 770)

(= (prioridadSuministroBase b6) 3)

(= (peticionPersonaBase b6) 968)

(= (prioridadPersonaBase b6) 2)

(= (peticionSuministroBase b7) 211)

(= (prioridadSuministroBase b7) 3)

(= (peticionPersonaBase b7) 347)

(= (prioridadPersonaBase b7) 1)

(= (personasEnCirculacion) 0)

(= (suministrosEnCirculacion) 0)

(= (total-cost) 0)

(= (penalizacion-total) 0)

)

(:goal

(and (forall (?b - base) (and (or (not(esAsentamiento ?b)) (= (personasEnBase ?b) 0)) (or (not(esAlmacen ?b)) (= (suministrosEnBase ?b) 0)))) (= (suministrosEnCirculacion) 0) (= (personasEnCirculacion) 0))

)

)

**Output:**

0: MOVERROVERDEBASE R1 B0 B1

1: MOVERROVERDEBASE R2 B1 B2

2: MOVERROVERDEBASE R3 B6 B7

3: MOVERROVERDEBASE R3 B7 B0

4: MOVERROVERDEBASE R3 B0 B1

5: MOVERROVERDEBASE R2 B2 B3

6: MOVERROVERDEBASE R1 B1 B2

7: CARGARPERSONA R2 B3

8: MOVERROVERDEBASE R2 B3 B4

9: DESCARGARPERSONA R2 B4

10: MOVERROVERDEBASE R1 B2 B3

11: MOVERROVERDEBASE R0 B5 B6

12: MOVERROVERDEBASE R0 B6 B7

13: MOVERROVERDEBASE R0 B7 B0

14: MOVERROVERDEBASE R0 B0 B1

15: MOVERROVERDEBASE R0 B1 B2

16: CARGARPERSONA R1 B3

17: CARGARPERSONA R1 B3

18: MOVERROVERDEBASE R1 B3 B4

19: DESCARGARPERSONA R1 B4

20: DESCARGARPERSONA R1 B4

21: MOVERROVERDEBASE R0 B2 B3

22: MOVERROVERDEBASE R4 B5 B6

23: MOVERROVERDEBASE R4 B6 B7

24: MOVERROVERDEBASE R4 B7 B0

25: MOVERROVERDEBASE R4 B0 B1

26: MOVERROVERDEBASE R3 B1 B2

27: CARGARPERSONA R0 B3

28: CARGARPERSONA R0 B3

29: MOVERROVERDEBASE R0 B3 B4

30: DESCARGARPERSONA R0 B4

31: DESCARGARPERSONA R0 B4

32: CARGARPERSONA R3 B2

33: CARGARPERSONA R3 B2

34: MOVERROVERDEBASE R3 B2 B3

35: MOVERROVERDEBASE R3 B3 B4

36: DESCARGARPERSONA R3 B4

37: DESCARGARPERSONA R3 B4

38: MOVERROVERDEBASE R4 B1 B2

39: MOVERROVERDEBASE R2 B4 B5

40: MOVERROVERDEBASE R2 B5 B6

41: MOVERROVERDEBASE R2 B6 B7

42: MOVERROVERDEBASE R2 B7 B0

43: MOVERROVERDEBASE R2 B0 B1

44: CARGARPERSONA R4 B2

45: MOVERROVERDEBASE R4 B2 B3

46: MOVERROVERDEBASE R4 B3 B4

47: DESCARGARPERSONA R4 B4

48: MOVERROVERDEBASE R2 B1 B2

49: MOVERROVERDEBASE R1 B4 B5

50: MOVERROVERDEBASE R1 B5 B6

51: MOVERROVERDEBASE R1 B6 B7

52: MOVERROVERDEBASE R1 B7 B0

53: MOVERROVERDEBASE R1 B0 B1

54: CARGARPERSONA R2 B2

55: MOVERROVERDEBASE R2 B2 B3

56: MOVERROVERDEBASE R2 B3 B4

57: DESCARGARPERSONA R2 B4

58: MOVERROVERDEBASE R1 B1 B2

59: MOVERROVERDEBASE R2 B4 B5

60: MOVERROVERDEBASE R2 B5 B6

61: MOVERROVERDEBASE R2 B6 B7

62: MOVERROVERDEBASE R2 B7 B0

63: MOVERROVERDEBASE R2 B0 B1

64: CARGARPERSONA R1 B2

65: MOVERROVERDEBASE R1 B2 B3

66: MOVERROVERDEBASE R1 B3 B4

67: DESCARGARPERSONA R1 B4

68: MOVERROVERDEBASE R2 B1 B2

69: MOVERROVERDEBASE R1 B4 B5

70: MOVERROVERDEBASE R1 B5 B6

71: MOVERROVERDEBASE R1 B6 B7

72: MOVERROVERDEBASE R1 B7 B0

73: MOVERROVERDEBASE R1 B0 B1

74: CARGARPERSONA R2 B2

75: MOVERROVERDEBASE R2 B2 B3

76: MOVERROVERDEBASE R2 B3 B4

77: DESCARGARPERSONA R2 B4

78: MOVERROVERDEBASE R1 B1 B2

79: MOVERROVERDEBASE R2 B4 B5

80: MOVERROVERDEBASE R2 B5 B6

81: MOVERROVERDEBASE R2 B6 B7

82: MOVERROVERDEBASE R2 B7 B0

83: MOVERROVERDEBASE R2 B0 B1

84: CARGARPERSONA R1 B2

85: CARGARPERSONA R1 B2

86: MOVERROVERDEBASE R1 B2 B3

87: MOVERROVERDEBASE R1 B3 B4

88: DESCARGARPERSONA R1 B4

89: DESCARGARPERSONA R1 B4

90: CARGARSUMINISTRO R2 B1

91: MOVERROVERDEBASE R0 B4 B5

92: MOVERROVERDEBASE R0 B5 B6

93: MOVERROVERDEBASE R2 B1 B2

94: MOVERROVERDEBASE R0 B6 B7

95: MOVERROVERDEBASE R2 B2 B3

96: MOVERROVERDEBASE R0 B7 B0

97: MOVERROVERDEBASE R2 B3 B4

98: MOVERROVERDEBASE R0 B0 B1

99: DESCARGARSUMINISTRO R2 B4

100: CARGARSUMINISTRO R0 B1

101: MOVERROVERDEBASE R4 B4 B5

102: MOVERROVERDEBASE R4 B5 B6

103: MOVERROVERDEBASE R0 B1 B2

104: MOVERROVERDEBASE R4 B6 B7

105: MOVERROVERDEBASE R0 B2 B3

106: MOVERROVERDEBASE R4 B7 B0

107: MOVERROVERDEBASE R0 B3 B4

108: MOVERROVERDEBASE R4 B0 B1

109: DESCARGARSUMINISTRO R0 B4

110: CARGARSUMINISTRO R4 B1

111: MOVERROVERDEBASE R4 B1 B2

112: MOVERROVERDEBASE R0 B4 B5

113: MOVERROVERDEBASE R0 B5 B6

114: MOVERROVERDEBASE R0 B6 B7

115: MOVERROVERDEBASE R0 B7 B0

116: MOVERROVERDEBASE R4 B2 B3

117: MOVERROVERDEBASE R0 B0 B1

118: MOVERROVERDEBASE R4 B3 B4

119: DESCARGARSUMINISTRO R4 B4

120: CARGARSUMINISTRO R0 B1

121: MOVERROVERDEBASE R0 B1 B2

122: MOVERROVERDEBASE R0 B2 B3

123: MOVERROVERDEBASE R0 B3 B4

124: DESCARGARSUMINISTRO R0 B4

El objetivo de este juego de pruebas sería observar como actua el planificador con una cantidad elevada de suministros y personal y si consigue generar un plan correctamente.

Como observamos lo ha hecho exitosamente.

## **6.2. Experimento tiempo de resolución**

En este experimento observamos como evoluciona el tiempo de resolución con el número de rovers. Lo que esperamos es que contra más rovers se tengan más trivial será el problema y por lo tanto menos tiempo de resolución tendrá.

Partimos diciéndole al generador que queremos 8 bases y no modificaremos este parámetro en ningún momento.

|  |  |
| --- | --- |
| Cantidad de rovers | Tiempo de ejecución |
| 2 | 0.12 |
| 4 | 0.79 |
| 7 | 1.42 |
| 10 | 3.73 |
| 12 | 2.26 |
| 14 | 7.68 |

Como observamos y a la inversa de lo que inicialmente pensabamos, el tiempo de ejecucción va aumentando poco a poco a medida que hay más rovers. Suponemos que esto es debido a que al planificador le produce una explosión combinatoria al unificar y puede hacer los mismos pasos de muchas formas distintas y lo explora todo.

Cuando superamos los 12 rovers vemos como el tiempo de ejecucción se dispara y con 15 rovers el planificador ya no puede generar un plan.