Práctica de Planificación

Menús RicoRico 2.0

INTELIGENCIA ARTIFICIAL
Grau en Enginyeria en Informàtica

Alejandro Gallego Rodríguez

Kamal El Hachmi

Vladislav Lapin

2020-2021 Q2

Facultat d'Informàtica de Barcelona
Universitat Politècnica de Catalunya

ÍNDICE

1. Introduccion

En este documento encontraremos la documentación de la práctica de Planificación del cuatrimestre de primavera 2020-2021, la cual consiste en la continuación del sistema de catering de la práctica anterior de SBC.

Esta vez se nos pide implementar un menú semanal, con ciertas restricciones sobre los platos en un día y sobre los menús de diferentes días.

Para resolver este problema hemos usado el conocimiento adquirido durante el curso y sobretodo en esta última parte. La estructura que hemos seguido en la resolución de este problema es la siguiente:

- **Identificación del problema:** Donde se justifican las restricciones, el conocimiento del sistema y el objetivo esperado.
- **Modelización del Dominio:** Donde dividimos el problema en partes, definimos las variables, las acciones, los predicados y las funciones .
- **Modelización de Problemas:** Donde definimos los objetos y sus características y los distintos estados del problema
- **Modelización:** Con todo lo que hemos descrito hasta ahora, trabajamos ya con código e implementamos todos los niveles de extensión de nuestro planificador.
- **Juegos de Prueba:** Para demostrar el correcto funcionamiento del programa presentamos unos juegos de prueba para cada extensión y explicamos el comportamiento del programa en cada caso.

2. Identificación del problema

La empresa Rico Rico, ahora quiere que el sistema además sea capaz de programar menús semanales (de lunes a viernes). Un menú está formado por un primero y un segundo, el cocinero, cada semana será capaz de preparar una lista de platos que tendremos que combinar y formar menús.

Evidentemente, hay restricciones obvias de incompatibilidad entre platos en un mismo menú, y además cada plato será de un tipo (sopa, ensalada, carne, ...) y habrá restricciones sobre no repetir el mismo tipo de plato en días consecutivos para primeros y para segundos.

También, el sistema conoce el valor calórico de cada plato y el sistema tiene que crear menús lo más equilibrados posibles. Para hacerlo más realista, cada plato tiene un tiempo de elaboración, nosotros en la práctica dos lo llamamos tiempo de cocción.

Para definir el programa de una manera más clara, definimos la entrada y la salida esperada del programa.

Al sistema le tenemos que dar una **lista de platos** (primeros y segundos), a cada plato asignarle una **tipología**, un **precio** y una **cantidad de calorías** y dar los datos de las **relaciones de incompatibilidad** entre primeros y segundos.

El resultado del sistema tiene que ser un menú que nos indicará qué platos preparar cada día de la semana.

3. Modelización del Dominio

En este apartado se describirá la modelización del dominio. Se detallarán las variables, predicados, acciones y funciones utilizadas para resolver esta práctica, junto una breve explicación para clarificar el sentido y uso de cada parte en el dominio.

3.1 Variables

- **plato:** tipo que indica que el objeto que se está utilizando es un plato
- **primero, segundo:** tipo que indica que el plato que se está utilizando es primero o segundo
- **dia:** tipo que indica que el objeto que se está utilizando es un día de la semana
- **tipo:** tipo que indica que el objeto que se está utilizando es un tipo de plato.

3.2 Predicados

- **(es_tipo ?p - plato ?t - tipo):** indica que el plato ?p es de tipo ?t
- **(incompatible ?p1 - plato ?p2 - plato):** indica el plato ?p1 es incompatible con el plato ?p2 y viceversa. Es decir, no pueden estar en el mismo menú de un día.
- **(menu_dia ?d - dia ?pr - primero ?sg - segundo):** indica que el dia ?d tiene de primer plato ?pr y de segundo ?sg.
- **(servido ?d - dia):** indica que el día ?d ya ha sido servido, es decir, ya tiene menú asignado.
- **(primero_usado ?p - primero):** indica que el primer plato ?p ya ha sido utilizado en otro menú con un día distinto.
- **(segundo_usado ?s - segundo):** indica que el segundo plato ?s ya ha sido utilizado en otro menú con un día distinto.
- **(tp_anterior ?tp - tipo):** indica qué tipo de primer plato ?tp ha sido utilizado el día anterior.
- **(ts_anterior ?ts - tipo):** indica qué tipo de segundo plato ?ts ha sido utilizado el día anterior.
- **(plato_dia ?d - dia ?p - plato):** indica que el dia ?d es obligatorio tener en su menú (primero o segundo) el plato ?p.

3.3 Acciones

- **preparar_menu(d, p, s, tp, ts):** para cada combinación de dia ***d***, primero ***p*** y de segundo ***s*** busca asignar el predicado ***(menu_dia ...)*** cumpliendo que no haya incompatibilidad entre ***p*** y ***s***, y que sean de tipo ***tp*** y ***ts***, respectivamente. Además de restricciones estipuladas en el enunciado, como por ejemplo, no tener un plato repetido en el menú semanal o no repetir tipo dos días seguidos.

3.4 Funciones

- **(cals-plato ?p - plato):** indica la cantidad de calorías que tiene el plato ?p.
- **(precio-plato ?p - plato):** indica el precio que cuesta el plato ?p.
- **(total-cost):** indica el coste total del menú semanal y busca minimizarlo.

4. Modelización de Problemas

4.1 Objetos

- **Primeros:** Se especifican todos los primeros que utilizaremos para modelar nuestro menú semanal. Como primeros hay:

Fideua_con_Sepia *Paella* *Crema_Calabacin* *Risotto_de_Pescado*
Huevos_Rellenos *Croquetas* *Canelones_de_Carne* *Ensalada_de_Col*
Tagliatelle_Bolonesa *Ramen* *Potaje_de_Alubias* *Patatas_Caliu*
Gazpacho *Pastel_de_Pan* *Spaghettis_Bolonesa* *Escudella*
Judias_Verdes *Cuscus* *Sushi* *Macarrones_Bolonesa*

- **Segundos:** Se especifican todos los segundos que utilizaremos para modelar nuestro menú semanal. Como segundos hay:

Sopa_de_Miso *Milanesa_Napolitana* *Costillas_de_Cerdo*
Canelones_de_Pollo *Tortilla_de_Espinacas* *Lasana_de_Atun* *Fingers_de_Pollo*
Berenjas_al_Horno *Merluza_a_Naranja* *Bacalao* *Suquet_de_Pescado*
Albondigas *Crema_Zanahoria* *Calamares_a_la_Romana* *Salmon_Ahumado*
Rape_al_Limon *Pastel_de_Verduras* *Calsotada* *Raviolli*
Falafel

- **Tipos:** Se especifican todos los tipos que existen en los platos especificados anteriormente. Como tipos hay:

Pescado *Carne* *Pasta* *Vegetal* *Sopa*

- **Dias:** Se especifican los días de la semana para crear los menús diarios. Como días hay:

Lunes *Martes* *Miercoles* *Jueves* *Viernes*

4.2 Estado Inicial

En el estado inicial debemos inicializar todos los predicados que son necesarios para empezar las iteraciones del problema, es decir aquellos predicados que sean prerrequisito de una acción y que no sean inicializados como efecto de la misma acción o de una distinta, en nuestro caso solo tenemos una acción.

Para cada primero y segundo que tengamos en objetos debemos asignarle un tipo. Para asignar un tipo necesitamos un plato y su correspondiente tipo, por ejemplo:

```
(es_tipo Fideua_con_Sepia Pescado)
(es_tipo Crema_Calabacin Vegetal)
(es_tipo Croquetas Carne)
```

Fideua_con_Sepia es de tipo Pescado, Crema_Calabacin es de tipo Vegetal y Croquetas es de tipo Carne.

También podemos inicializar las incompatibilidades, esto lo veremos más adelante en los siguientes apartados con más detalle. Las incompatibilidades se inicializan con dos platos distintos ya sea primero o segundo. Un ejemplo podría ser:

```
(incompatible Tagliatelle_Bolonesa Bacalao)
(incompatible Judias_Verdes Pastel_de_Verduras)
```

Tagliatelle_Bolonesa es incompatible con el Bacalao y Judias_Verdes es incompatible con el Pastel_de_Verduras, por lo tanto no se podrá construir ningún menú diario que contenga ambos platos juntos.

Para finalizar con la inicialización de predicados, es totalmente necesario inicializar aquello que utilizaremos para saber si hemos llegado al estado final. Utilizaremos el predicado de servido para comprobar que a un día ya le ha sido asignado un menú, pero primero debemos indicarle que no se le ha asignado ningún menú para eso haremos en todos nuestros problemas:

```
(not (servido Viernes))
(not (servido Jueves))
(not (servido Miercoles))
(not (servido Martes))
(not (servido Lunes))
```

Una vez acabados con los predicados empezaremos con las funciones. Cada plato debe tener un precio y la cantidad de calorías que contiene, para así poder completar las extensiones 4 y 5. Para inicializar estas funciones debemos utilizar Metric-FF ya que no podemos crear predicados y asignarles un número cualquiera sin el requerimiento *:fluents*.

Entonces para cada primero y segundo se le dará un valor que en el siguiente apartado comentaremos de donde sale. Además debemos inicializar el coste total a 0 para utilizarlo de sumatorio y minimizarlo:

```
(= (precio-plato Falafel) 9)
(= (cals-plato Falafel) 781)
     (= (total-cost) 0)
```

4.3 Estado final

Llegaremos al estado final, si y sólo si, se cumple la condición estipulada en el *:goal*, esta condición es la de que todos los platos hayan sido servidos. Hemos explicado antes que el predicado *(servido ?d - dia)* debía ser inicializado a false siempre y es por esta misma razón. El efecto de la acción, además de otras cosas, inicializa este predicado con el día con el que acaba de preparar el menú. Cuando los 5 días tengan este predicado iniciado a true, las iteraciones acabarán y se desplegará la traza con los menús escogidos.

No solo eso si no que para la extensión 5 es necesario minimizar el coste total del menú semanal y por lo tanto utilizaremos Metric-FF para hacer uso de la acción *:metric* que nos permite minimizar el coste total.

```
(:goal (forall (?d - dia) (servido ?d)))
     (:metric minimize (total-cost))
```

5. Modelización

Los modelos se han ido desarrollando por iteraciones siendo cada iteración una extensión del problema. Para cada iteración se ha añadido tanto en el dominio como en el problema los elementos que hacían falta para completar dicha extensión.

5.1 Nivel básico

En esta parte de la modelización estaban definidos los tipos de forma definitiva, a excepción de la variable *tipo* y teníamos tres predicados los cuales eran *(incompatible ?p1 - plato ?p2 - plato), (servido ?d - dia)* y *(menu_dia ?d - dia ?p - primero ?s - segundo),* con estos predicados nos era suficiente para superar el nivel básico ya que era la precondición de la acción, que en ese momento contenía 3 parámetros (dia, primero y segundo) comprobaba que primero y segundo fueran compatibles y que el dia no estuviera servido. Con esto el efecto de la acción consistía en servir el día y asignar el predicado *(menu_dia ...)* y con ello llegar al estado final.

```
:parameters (?d - dia ?p - primero ?s - segundo)
:precondition ( and
            ..
            (not (incompatible ?p ?s)) (not (servido ?d))
            ..)
:effect (and
      ..
      (servido ?d) (menu_dia ?d ?p ?s)
      ..
)
```

5.2 Extensión 1

En la extensión 1 debíamos controlar que los platos no se repitieran durante la semana, es por eso que decidimos añadir dos predicados: *(primero_usado ?p - primero)* y *(segundo_usado ?s - segundo),* con estos predicados podíamos controlar que no se repitieran platos simplemente negándolo en la precondición y posteriormente inicializandolo en el efecto de la acción con los correspondientes platos.

```
:parameters (..)
:precondition ( and
           ..
         (not (primero_usado ?p)) (not (segundo_usado ?s))
           ..)
:effect (and
      ..
      (primero_usado ?p) (segundo_usado ?s)
      ..
)
```

5.3 Extensión 2

En la extensión 2 nos pedían introducir los tipos de platos y controlarlos de forma que no se repitieran primeros de un mismo tipo en días consecutivos, asimismo con los segundos. Para conseguir esto introducimos la variable *tipo* y los predicados *(tp_anterior ?tp - tipo)* y *(ts_anterior ?ts - tipo),* además de añadir dos nuevos parámetros (tipo segundo y tipo primero) a la acción. Consiguiendo así controlar que tipo había sido en el día anterior y evitarlo en forma de precondición y posteriormente inicializarlo en el efecto de la acción para preparar el menú del próximo día.

```
:parameters (?d - dia ?p - primero ?s - segundo ?tp - tipo ?ts -
tipo)
:precondition ( and
           ..
           (es_tipo ?p ?tp) (es_tipo ?s ?ts)
           ..)
:effect (and
      ..
      (tp_anterior ?tp) (ts_anterior ?ts)
      ..)
```

5.4 Extensión 3

En la extensión 3 debíamos introducir la opción de que hubieran días especiales con un plato que obligatoriamente debía estar. Para llevar a cabo esto, se añadió un predicado más, el (*plato_dia* ?d - dia ?p - plato) y modificar la precondición de la acción "preparar_menu" añadiendo "`(or (plato_dia ?d ?p) (plato_dia ?d ?s)...)`" para tratar lo antes posible los platos específicos. Consiguiendo así el objetivo de tener un plato específico en un día concreto

```
(:action preparar_menu
        :parameters (?d - dia ?p - primero ?s - segundo ?tp - tipo
?ts - tipo)
        :precondition (or
            (plato_dia ?d ?p) (plato_dia ?d ?s)
            (and
            ..))
         :effect(and
           ..)
```

5.5 Extensión 4

En la penúltima extensión empezamos a utilizar Metric-FF y con ello añadimos el requerimiento *:fluents* y la función *(cals-plato ?p - plato)* para así poder operar con enteros y tener una referencia de las calorías que tiene determinado plato. Este entero nos serviría para añadir una precondición utilizando la suma de ambós platos y comprobar que está en el intervalo de 1000 a 1500:

```
:parameters (..)
:precondition ( and
           ..
           (and
           (<= (+ (cals-plato ?p) (cals-plato?s)) 1500)
           (>= (+ (cals-plato ?p) (cals-plato?s)) 1000))
           ..)
:effect (and
     ..
)
```

5.6 Extensión 5

En la última extensión debemos minimizar el coste total del menú semanal, es por eso que introducimos dos nuevas funciones: *(precio-plato ?p - plato)* y *(total-cost)*. Estas funciones deben ser inicializadas en el fichero de problemas y utilizamos el efecto de la acción para hacer un sumatorio a total-cost con los precios del primer y segundo plato. Finalmente en el estado final utilizamos *:metric* para minimizar el coste total del menú.

```
:parameters (..)
:precondition ( and
           ..
           )
:effect (and
      ..
      (increase (total-cost) (precio-plato ?p))
      (increase (total-cost) (precio-plato ?s)))
)
```

6. Juegos de prueba

En este apartado comentaremos los juegos de prueba utilizados en cada extensión para comprobar el correcto funcionamiento del planificador. Es necesario decir que cada juego de prueba realizado ha sido generado automáticamente con un script en Python.

Este script contiene 2 vectores con todos los platos disponibles en nuestro planificador, que entre primeros y segundos son 40 en total. Cada posición de este vector indica el nombre del plato y su tipo correspondiente siendo este un vector de vectores. Podríamos haber utilizado un HashMap pero por temas de simplicidad decidimos irnos por la matriz.

Al principio del script inicializa dos variables con un número random entre 15 y 20 indicando el número de primeros y segundos que se escogerán. Se utiliza random.sample en cada vector para así tener la mayor aleatoriedad posible y dar resultados variables. La otra variable se inicializa con un número, también, random entre 0 y 10 e indica el número de incompatibilidades tendremos. Gracias a esto cada juego de pruebas generado es totalmente diferente al anterior. Las calorías y el precio de cada plato también se genera aleatoriamente entre 250 y 750, y 5 y 10 respectivamente para los primeros y entre 400 y 900, y 8 y 15 para los segundos. Dándonos así el máximo grado de aleatoriedad posible. Utilizamos también aleatoriedad para decidir los platos del dia, computando un numero random entre 0 y 2 para decidir el número de platos del día, además de decidir también aleatoriamente el plato y el día del conjunto de platos.

Todos los juegos de pruebas que se muestran a continuación han sido generados a partir del generador de problemas.

6.1 Nivel Básico

En este nivel básico debemos generar un menú sin incompatibilidades, es decir, no deben aparecer las combinaciones que se especifican en el predicado de incompatible, prevemos que salga el mismo menú para todos los días.

➢ 1er Juego de Pruebas

Problema:

```
(define (problem catering-15platos-3incmpls)
    (:domain catering)
    (:objects Pastel_de_Pan Spaghettis_Bolonesa Risotto_de_Pescado
Potaje_de_Alubias Sushi Crema_Calabacin Cuscus Patatas_Caliu
Tagliatelle_Bolonesa Croquetas Gazpacho Fideua_con_Sepia Macarrones_Bolonesa
Judias_Verdes Canelones_de_Carne - primero
              Pastel_de_Verduras Fingers_de_Pollo Falafel Albondigas
Rape_al_Limon Salmon_Ahumado Calsotada Costillas_de_Cerdo Crema_Zanahoria
Lasana_de_Atun Sopa_de_Miso Milanesa_Napolitana Bacalao Calamares_a_la_Romana
Raviolli - segundo
              Lunes Martes Miercoles Jueves Viernes - dia
    )
    (:init
       (incompatible Fideua_con_Sepia Pastel_de_Verduras)
       (incompatible Potaje_de_Alubias Crema_Zanahoria)
       (incompatible Canelones_de_Carne Rape_al_Limon)
       (not (servido Viernes))
       (not (servido Jueves))
       (not (servido Miercoles))
       (not (servido Martes))
       (not (servido Lunes))
    )
    (:goal (
         forall (?d - dia) (servido ?d)
        )
    )
)
```

Resultado:

step 0: PREPARAR_MENU LUNES CANELONES_DE_CARNE SOPA_DE_MISO
1: PREPARAR_MENU MARTES CANELONES_DE_CARNE SOPA_DE_MISO
2: PREPARAR_MENU MIERCOLES CANELONES_DE_CARNE SOPA_DE_MISO
3: PREPARAR_MENU JUEVES CANELONES_DE_CARNE SOPA_DE_MISO
4: PREPARAR_MENU VIERNES SUSHI CANELONES_DE_CARNE SOPA_DE_MISO

➢ 2do Juego de Pruebas

Problema:

```
(define (problem catering-16platos-9incmpls)
    (:domain catering)
    (:objects Cuscus Canelones_de_Carne Potaje_de_Alubias Croquetas Gazpacho
Judias_Verdes Risotto_de_Pescado Paella Sushi Ensalada_de_Col Escudella Ramen
Huevos_Rellenos Macarrones_Bolonesa Pastel_de_Pan Crema_Calabacin - primero
              Canelones_de_Pollo Raviolli Costillas_de_Cerdo
Calamares_a_la_Romana Milanesa_Napolitana Tortilla_de_Espinacas
Fingers_de_Pollo Suquet_de_Pescado Crema_Zanahoria Lasana_de_Atun
Rape_al_Limon Salmon_Ahumado Albondigas Calsotada Sopa_de_Miso
Merluza_a_Naranja - segundo
              Lunes Martes Miercoles Jueves Viernes - dia
    )
    (:init
       (incompatible Gazpacho Rape_al_Limon)
       (incompatible Escudella Calsotada)
       (incompatible Ramen Costillas_de_Cerdo)
       (incompatible Ensalada_de_Col Calamares_a_la_Romana)
       (incompatible Risotto_de_Pescado Fingers_de_Pollo)
       (incompatible Canelones_de_Carne Salmon_Ahumado)
       (incompatible Judias_Verdes Sopa_de_Miso)
       (incompatible Cuscus Raviolli)
       (incompatible Huevos_Rellenos Milanesa_Napolitana)
       (not (servido Viernes))
       (not (servido Jueves))
       (not (servido Miercoles))
       (not (servido Martes))
       (not (servido Lunes))
    )
    (:goal (
         forall (?d - dia) (servido ?d)
        )
    )
)
```

Resultado:

step 0: PREPARAR_MENU LUNES CREMA_CALABACIN SOPA_DE_MISO
1: PREPARAR_MENU MARTES CREMA_CALABACIN SOPA_DE_MISO
2: PREPARAR_MENU MIERCOLES CREMA_CALABACIN SOPA_DE_MISO
3: PREPARAR_MENU JUEVES CREMA_CALABACIN SOPA_DE_MISO
4: PREPARAR_MENU VIERNES CREMA_CALABACIN SOPA_DE_MISO

6.2 Extension 1

Utilizaremos los mismos juegos de prueba anteriores ya que no cambiamos nada en el problema, solo cambiamos en el dominio. Deberían salir en el resultado diferentes combinaciones de platos para cada día siguiendo sin haber incompatibilidades.

➢ 1er Juego de Prueba

Problema:

```
(define (problem catering-15platos-3incmpls)
    (:domain catering)
    (:objects Pastel_de_Pan Spaghettis_Bolonesa Risotto_de_Pescado
Potaje_de_Alubias Sushi Crema_Calabacin Cuscus Patatas_Caliu
Tagliatelle_Bolonesa Croquetas Gazpacho Fideua_con_Sepia Macarrones_Bolonesa
Judias_Verdes Canelones_de_Carne - primero
              Pastel_de_Verduras Fingers_de_Pollo Falafel Albondigas
Rape_al_Limon Salmon_Ahumado Calsotada Costillas_de_Cerdo Crema_Zanahoria
Lasana_de_Atun Sopa_de_Miso Milanesa_Napolitana Bacalao Calamares_a_la_Romana
Raviolli - segundo
              Lunes Martes Miercoles Jueves Viernes - dia
    )
    (:init
       (incompatible Fideua_con_Sepia Pastel_de_Verduras)
       (incompatible Potaje_de_Alubias Crema_Zanahoria)
       (incompatible Canelones_de_Carne Rape_al_Limon)
       (not (servido Viernes))
       (not (servido Jueves))
       (not (servido Miercoles))
       (not (servido Martes))
       (not (servido Lunes))
    )
    (:goal (
         forall (?d - dia) (servido ?d)
        )
    )
)
```

Resultado:

step 0: PREPARAR_MENU LUNES CANELONES_DE_CARNE SOPA_DE_MISO
1: PREPARAR_MENU MARTES JUDIAS_VERDES CALSOTADA
2: PREPARAR_MENU MIERCOLES MACARRONES_BOLONESA RAVIOLLI
3: PREPARAR_MENU JUEVES FIDEUA_CON_SEPIA MILANESA_NAPOLITANA
4: PREPARAR_MENU VIERNES GAZPACHO CALAMARES_A_LA_ROMANA

➢ 2do Juego de Pruebas

Problema:

```
(define (problem catering-16platos-9incmpls)
    (:domain catering)
    (:objects Cuscus Canelones_de_Carne Potaje_de_Alubias Croquetas Gazpacho
Judias_Verdes Risotto_de_Pescado Paella Sushi Ensalada_de_Col Escudella Ramen
Huevos_Rellenos Macarrones_Bolonesa Pastel_de_Pan Crema_Calabacin - primero
              Canelones_de_Pollo Raviolli Costillas_de_Cerdo
Calamares_a_la_Romana Milanesa_Napolitana Tortilla_de_Espinacas
Fingers_de_Pollo Suquet_de_Pescado Crema_Zanahoria Lasana_de_Atun
Rape_al_Limon Salmon_Ahumado Albondigas Calsotada Sopa_de_Miso
Merluza_a_Naranja - segundo
              Lunes Martes Miercoles Jueves Viernes - dia
    )
    (:init
       (incompatible Gazpacho Rape_al_Limon)
       (incompatible Escudella Calsotada)
       (incompatible Ramen Costillas_de_Cerdo)
       (incompatible Ensalada_de_Col Calamares_a_la_Romana)
       (incompatible Risotto_de_Pescado Fingers_de_Pollo)
       (incompatible Canelones_de_Carne Salmon_Ahumado)
       (incompatible Judias_Verdes Sopa_de_Miso)
       (incompatible Cuscus Raviolli)
       (incompatible Huevos_Rellenos Milanesa_Napolitana)
       (not (servido Viernes))
       (not (servido Jueves))
       (not (servido Miercoles))
       (not (servido Martes))
       (not (servido Lunes))
    )
    (:goal (
         forall (?d - dia) (servido ?d)
        )
    )
)
```

Resultado:

step 0: PREPARAR_MENU LUNES CREMA_CALABACIN SOPA_DE_MISO
1: PREPARAR_MENU MARTES MACARRONES_BOLONESA CALSOTADA
2: PREPARAR_MENU MIERCOLES HUEVOS_RELLENOS RAVIOLLI
3: PREPARAR_MENU JUEVES RAMEN ALBONDIGAS
4: PREPARAR_MENU VIERNES SUSHI MERLUZA_A_NARANJA

6.3 Extension 2

Añadiremos a los juegos de prueba anteriores los predicados necesarios en los problemas para poder representar el tipo de cada plato. Deberían salir combinaciones de platos que no tengan el mismo tipo en días consecutivos.

➢ 1er Juego de Pruebas

Problema:

```
(define (problem catering-15platos-3incmpls)
    (:domain catering)
    (:objects Pastel_de_Pan Spaghettis_Bolonesa Risotto_de_Pescado
Potaje_de_Alubias Sushi Crema_Calabacin Cuscus Patatas_Caliu
Tagliatelle_Bolonesa Croquetas Gazpacho Fideua_con_Sepia Macarrones_Bolonesa
Judias_Verdes Canelones_de_Carne - primero
              Pastel_de_Verduras Fingers_de_Pollo Falafel Albondigas
Rape_al_Limon Salmon_Ahumado Calsotada Costillas_de_Cerdo Crema_Zanahoria
Lasana_de_Atun Sopa_de_Miso Milanesa_Napolitana Bacalao Calamares_a_la_Romana
Raviolli - segundo
              Pescado Carne Pasta Vegetal Sopa - tipo
              Lunes Martes Miercoles Jueves Viernes - dia
    )
    (:init
       (es_tipo Pastel_de_Pan Vegetal)
       (es_tipo Spaghettis_Bolonesa Pasta)
       (es_tipo Risotto_de_Pescado Pescado)
       (es_tipo Potaje_de_Alubias Sopa)
       (es_tipo Sushi Pescado)
       (es_tipo Crema_Calabacin Vegetal)
       (es_tipo Cuscus Vegetal)
       (es_tipo Patatas_Caliu Vegetal)
       (es_tipo Tagliatelle_Bolonesa Pasta)
       (es_tipo Croquetas Carne)
       (es_tipo Gazpacho Sopa)
       (es_tipo Fideua_con_Sepia Pescado)
       (es_tipo Macarrones_Bolonesa Pasta)
       (es_tipo Judias_Verdes Vegetal)
       (es_tipo Canelones_de_Carne Carne)
       (es_tipo Pastel_de_Verduras Vegetal)
       (es_tipo Fingers_de_Pollo Carne)
       (es_tipo Falafel Carne)
       (es_tipo Albondigas Carne)
       (es_tipo Rape_al_Limon Pescado)
       (es_tipo Salmon_Ahumado Pescado)
       (es_tipo Calsotada Vegetal)
       (es_tipo Costillas_de_Cerdo Carne)
       (es_tipo Crema_Zanahoria Sopa)
       (es_tipo Lasana_de_Atun Pescado)
       (es_tipo Sopa_de_Miso Sopa)
```

```
        (es_tipo Milanesa_Napolitana Carne)
        (es_tipo Bacalao Pescado)
        (es_tipo Calamares_a_la_Romana Pescado)
        (es_tipo Raviolli Pasta)
        (incompatible Fideua_con_Sepia Pastel_de_Verduras)
        (incompatible Potaje_de_Alubias Crema_Zanahoria)
        (incompatible Canelones_de_Carne Rape_al_Limon)
        (not (servido Viernes))
        (not (servido Jueves))
        (not (servido Miercoles))
        (not (servido Martes))
        (not (servido Lunes))
    )
    (:goal (
         forall (?d - dia) (servido ?d)
        )
    )
)
```

Resultado:

step 0: PREPARAR_MENU LUNES CANELONES_DE_CARNE SOPA_DE_MISO CARNE SOPA
1: PREPARAR_MENU MARTES JUDIAS_VERDES CALSOTADA VEGETAL VEGETAL
2: PREPARAR_MENU MIERCOLES MACARRONES_BOLONESA RAVIOLLI PASTA PASTA
3: PREPARAR_MENU JUEVES FIDEUA_CON_SEPIA MILANESA_NAPOLITANA PESCADO CARNE
4: PREPARAR_MENU VIERNES GAZPACHO CALAMARES_A_LA_ROMANA SOPA PESCADO

➢ 2do Juego de Pruebas

Problema:

```
(define (problem catering-16platos-9incmpls)
    (:domain catering)
    (:objects Cuscus Canelones_de_Carne Potaje_de_Alubias Croquetas Gazpacho
Judias_Verdes Risotto_de_Pescado Paella Sushi Ensalada_de_Col Escudella Ramen
Huevos_Rellenos Macarrones_Bolonesa Pastel_de_Pan Crema_Calabacin - primero
              Canelones_de_Pollo Raviolli Costillas_de_Cerdo
Calamares_a_la_Romana Milanesa_Napolitana Tortilla_de_Espinacas
Fingers_de_Pollo Suquet_de_Pescado Crema_Zanahoria Lasana_de_Atun
Rape_al_Limon Salmon_Ahumado Albondigas Calsotada Sopa_de_Miso
Merluza_a_Naranja - segundo
              Pescado Carne Pasta Vegetal Sopa - tipo
              Viernes Jueves Miercoles Martes Lunes - dia
    )
    (:init
       (es_tipo Cuscus Vegetal)
       (es_tipo Canelones_de_Carne Carne)
       (es_tipo Potaje_de_Alubias Sopa)
       (es_tipo Croquetas Carne)
       (es_tipo Gazpacho Sopa)
       (es_tipo Judias_Verdes Vegetal)
       (es_tipo Risotto_de_Pescado Pescado)
       (es_tipo Paella Pescado)
       (es_tipo Sushi Pescado)
       (es_tipo Ensalada_de_Col Vegetal)
       (es_tipo Escudella Sopa)
       (es_tipo Ramen Sopa)
       (es_tipo Huevos_Rellenos Carne)
       (es_tipo Macarrones_Bolonesa Pasta)
       (es_tipo Pastel_de_Pan Vegetal)
       (es_tipo Crema_Calabacin Vegetal)
       (es_tipo Canelones_de_Pollo Carne)
       (es_tipo Raviolli Pasta)
       (es_tipo Costillas_de_Cerdo Carne)
       (es_tipo Calamares_a_la_Romana Pescado)
       (es_tipo Milanesa_Napolitana Carne)
       (es_tipo Tortilla_de_Espinacas Vegetal)
       (es_tipo Fingers_de_Pollo Carne)
       (es_tipo Suquet_de_Pescado Sopa)
       (es_tipo Crema_Zanahoria Sopa)
       (es_tipo Lasana_de_Atun Pescado)
       (es_tipo Rape_al_Limon Pescado)
       (es_tipo Salmon_Ahumado Pescado)
       (es_tipo Albondigas Carne)
       (es_tipo Calsotada Vegetal)
       (es_tipo Sopa_de_Miso Sopa)
       (es_tipo Merluza_a_Naranja Pescado)
       (incompatible Gazpacho Rape_al_Limon)
       (incompatible Escudella Calsotada)
```

```
        (incompatible Ramen Costillas_de_Cerdo)
        (incompatible Ensalada_de_Col Calamares_a_la_Romana)
        (incompatible Risotto_de_Pescado Fingers_de_Pollo)
        (incompatible Canelones_de_Carne Salmon_Ahumado)
        (incompatible Judias_Verdes Sopa_de_Miso)
        (incompatible Cuscus Raviolli)
        (incompatible Huevos_Rellenos Milanesa_Napolitana)
        (not (servido Viernes))
        (not (servido Jueves))
        (not (servido Miercoles))
        (not (servido Martes))
        (not (servido Lunes))
    )
    (:goal (
         forall (?d - dia) (servido ?d)
        )
    )
)
```

Resultado:

step 0: PREPARAR_MENU VIERNES CREMA_CALABACIN SOPA_DE_MISO VEGETAL SOPA
1: PREPARAR_MENU JUEVES MACARRONES_BOLONESA CALSOTADA PASTA VEGETAL
2: PREPARAR_MENU MIERCOLES HUEVOS_RELLENOS RAVIOLLI CARNE PASTA
3: PREPARAR_MENU MARTES RAMEN ALBONDIGAS SOPA CARNE
4: PREPARAR_MENU LUNES SUSHI MERLUZA_A_NARANJA PESCADO PESCADO

6.4 Extension 3

Para llevar a cabo el correcto funcionamiento de la extensión 3, hemos reutilizado los juegos de prueba del apartado anterior (6.3), añadiendoles diferentes relaciones/predicados de "plato específico" (`plato_dia`) en la inicialización. En el resultado se comprueba que deben de salir combinaciones de platos que tengan los platos específicos para un día determinado.

➢ 1er Juego de Pruebas

Problema:

```
(define (problem catering-15platos-3incmpls)
    (:domain catering)
    (:objects Pastel_de_Pan Spaghettis_Bolonesa Risotto_de_Pescado
Potaje_de_Alubias Sushi Crema_Calabacin Cuscus Patatas_Caliu
Tagliatelle_Bolonesa Croquetas Gazpacho Fideua_con_Sepia Macarrones_Bolonesa
Judias_Verdes Canelones_de_Carne - primero
              Pastel_de_Verduras Fingers_de_Pollo Falafel Albondigas
Rape_al_Limon Salmon_Ahumado Calsotada Costillas_de_Cerdo Crema_Zanahoria
Lasana_de_Atun Sopa_de_Miso Milanesa_Napolitana Bacalao Calamares_a_la_Romana
Raviolli - segundo
              Pescado Carne Pasta Vegetal Sopa - tipo
              Lunes Martes Miercoles Jueves Viernes - dia
    )
    (:init
       (es_tipo Pastel_de_Pan Vegetal)
       (es_tipo Spaghettis_Bolonesa Pasta)
       (es_tipo Risotto_de_Pescado Pescado)
       (es_tipo Potaje_de_Alubias Sopa)
       (es_tipo Sushi Pescado)
       (es_tipo Crema_Calabacin Vegetal)
       (es_tipo Cuscus Vegetal)
       (es_tipo Patatas_Caliu Vegetal)
       (es_tipo Tagliatelle_Bolonesa Pasta)
       (es_tipo Croquetas Carne)
       (es_tipo Gazpacho Sopa)
       (es_tipo Fideua_con_Sepia Pescado)
       (es_tipo Macarrones_Bolonesa Pasta)
       (es_tipo Judias_Verdes Vegetal)
       (es_tipo Canelones_de_Carne Carne)
       (es_tipo Pastel_de_Verduras Vegetal)
       (es_tipo Fingers_de_Pollo Carne)
       (es_tipo Falafel Carne)
       (es_tipo Albondigas Carne)
       (es_tipo Rape_al_Limon Pescado)
       (es_tipo Salmon_Ahumado Pescado)
       (es_tipo Calsotada Vegetal)
       (es_tipo Costillas_de_Cerdo Carne)
```

```
        (es_tipo Crema_Zanahoria Sopa)
        (es_tipo Lasana_de_Atun Pescado)
        (es_tipo Sopa_de_Miso Sopa)
        (es_tipo Milanesa_Napolitana Carne)
        (es_tipo Bacalao Pescado)
        (es_tipo Calamares_a_la_Romana Pescado)
        (es_tipo Raviolli Pasta)
        (incompatible Fideua_con_Sepia Pastel_de_Verduras)
        (incompatible Potaje_de_Alubias Crema_Zanahoria)
        (incompatible Canelones_de_Carne Rape_al_Limon)
        (plato_dia Jueves Gazpacho)
        (plato_dia Martes Calsotada)
        (plato_dia Lunes Crema_Zanahoria)
        (not (servido Viernes))
        (not (servido Jueves))
        (not (servido Miercoles))
        (not (servido Martes))
        (not (servido Lunes))
    )
    (:goal (
         forall (?d - dia) (servido ?d)
        )
    )
)
```

Resultado:

step 0: PREPARAR_MENU JUEVES GAZPACHO COSTILLAS_DE_CERDO SOPA VEGETAL
1: PREPARAR_MENU MARTES SUSHI CALSOTADA VEGETAL PESCADO
2: PREPARAR_MENU LUNES PASTEL_DE_PAN CREMA_ZANAHORIA PESCADO VEGETAL
3: PREPARAR_MENU MIERCOLES MACARRONES_BOLONESA RAVIOLLI PASTA PASTA
4: PREPARAR_MENU VIERNES CROQUETAS FINGERS_DE_POLLO CARNE CARNE

➢ 2do Juego de Pruebas

Problema:

```
(define (problem catering-16platos-9incmpls)
    (:domain catering)
    (:objects Cuscus Canelones_de_Carne Potaje_de_Alubias Croquetas Gazpacho
Judias_Verdes Risotto_de_Pescado Paella Sushi Ensalada_de_Col Escudella Ramen
Huevos_Rellenos Macarrones_Bolonesa Pastel_de_Pan Crema_Calabacin - primero
              Canelones_de_Pollo Raviolli Costillas_de_Cerdo
Calamares_a_la_Romana Milanesa_Napolitana Tortilla_de_Espinacas
Fingers_de_Pollo Suquet_de_Pescado Crema_Zanahoria Lasana_de_Atun
Rape_al_Limon Salmon_Ahumado Albondigas Calsotada Sopa_de_Miso
Merluza_a_Naranja - segundo
              Pescado Carne Pasta Vegetal Sopa - tipo
              Viernes Jueves Miercoles Martes Lunes - dia
    )
    (:init
       (es_tipo Cuscus Vegetal)
       (es_tipo Canelones_de_Carne Carne)
       (es_tipo Potaje_de_Alubias Sopa)
       (es_tipo Croquetas Carne)
       (es_tipo Gazpacho Sopa)
       (es_tipo Judias_Verdes Vegetal)
       (es_tipo Risotto_de_Pescado Pescado)
       (es_tipo Paella Pescado)
       (es_tipo Sushi Pescado)
       (es_tipo Ensalada_de_Col Vegetal)
       (es_tipo Escudella Sopa)
       (es_tipo Ramen Sopa)
       (es_tipo Huevos_Rellenos Carne)
       (es_tipo Macarrones_Bolonesa Pasta)
       (es_tipo Pastel_de_Pan Vegetal)
       (es_tipo Crema_Calabacin Vegetal)
       (es_tipo Canelones_de_Pollo Carne)
       (es_tipo Raviolli Pasta)
       (es_tipo Costillas_de_Cerdo Carne)
       (es_tipo Calamares_a_la_Romana Pescado)
       (es_tipo Milanesa_Napolitana Carne)
       (es_tipo Tortilla_de_Espinacas Vegetal)
       (es_tipo Fingers_de_Pollo Carne)
       (es_tipo Suquet_de_Pescado Sopa)
       (es_tipo Crema_Zanahoria Sopa)
       (es_tipo Lasana_de_Atun Pescado)
       (es_tipo Rape_al_Limon Pescado)
       (es_tipo Salmon_Ahumado Pescado)
       (es_tipo Albondigas Carne)
       (es_tipo Calsotada Vegetal)
       (es_tipo Sopa_de_Miso Sopa)
       (es_tipo Merluza_a_Naranja Pescado)
       (incompatible Gazpacho Rape_al_Limon)
       (incompatible Escudella Calsotada)
```

```
        (incompatible Ramen Costillas_de_Cerdo)
        (incompatible Ensalada_de_Col Calamares_a_la_Romana)
        (incompatible Risotto_de_Pescado Fingers_de_Pollo)
        (incompatible Canelones_de_Carne Salmon_Ahumado)
        (incompatible Judias_Verdes Sopa_de_Miso)
        (incompatible Cuscus Raviolli)
        (incompatible Huevos_Rellenos Milanesa_Napolitana)
        (plato_dia Jueves Paella)
        (not (servido Viernes))
        (not (servido Jueves))
        (not (servido Miercoles))
        (not (servido Martes))
        (not (servido Lunes))
    )
    (:goal (
         forall (?d - dia) (servido ?d)
        )
    )
)
```

Resultado:

step 0: PREPARAR_MENU JUEVES PAELLA CANELONES_DE_POLLO PESCADO PASTA
1: PREPARAR_MENU VIERNES CROQUETAS LASANA_DE_ATUN CARNE PESCADO
2: PREPARAR_MENU MIERCOLES MACARRONES_BOLONESA SUQUET_DE_PESCADO PASTA SOPA
3: PREPARAR_MENU MARTES ESCUDELLA TORTILLA_DE_ESPINACAS SOPA VEGETAL
4: PREPARAR_MENU LUNES CUSCUS COSTILLAS_DE_CERDO VEGETAL CARNE

6.5 Extension 4

Seguiremos con los juegos de pruebas anteriores, excluyendo los platos del día para así tener una solución más variada, donde añadiremos en el problema las calorías de cada plato para así calcular que cada menú este entre 1000 y 1500 calorías. Por temas de espacio seguiremos en la casilla init con lo añadido para esta extensión. Para estos juegos de prueba esperamos menús que están dentro del intervalo de calorías.

➢ 1er Juego de Prueba

Problema:

```
(define (problem catering-15platos-3incmpls)
    (:domain catering)
    (:objects Pastel_de_Pan Spaghettis_Bolonesa Risotto_de_Pescado
Potaje_de_Alubias Sushi Crema_Calabacin Cuscus Patatas_Caliu
Tagliatelle_Bolonesa Croquetas Gazpacho Fideua_con_Sepia Macarrones_Bolonesa
Judias_Verdes Canelones_de_Carne - primero
              Pastel_de_Verduras Fingers_de_Pollo Falafel Albondigas
Rape_al_Limon Salmon_Ahumado Calsotada Costillas_de_Cerdo Crema_Zanahoria
Lasana_de_Atun Sopa_de_Miso Milanesa_Napolitana Bacalao Calamares_a_la_Romana
Raviolli - segundo
              Pescado Carne Pasta Vegetal Sopa - tipo
              Lunes Martes Miercoles Jueves Viernes - dia
    )
    (:init
      ..
       (= (cals-plato Pastel_de_Pan) 348)
       (= (cals-plato Spaghettis_Bolonesa) 352)
       (= (cals-plato Risotto_de_Pescado) 330)
       (= (cals-plato Potaje_de_Alubias) 413)
       (= (cals-plato Sushi) 725)
       (= (cals-plato Crema_Calabacin) 445)
       (= (cals-plato Cuscus) 582)
       (= (cals-plato Patatas_Caliu) 613)
       (= (cals-plato Tagliatelle_Bolonesa) 393)
       (= (cals-plato Croquetas) 410)
       (= (cals-plato Gazpacho) 693)
       (= (cals-plato Fideua_con_Sepia) 574)
       (= (cals-plato Macarrones_Bolonesa) 580)
       (= (cals-plato Judias_Verdes) 272)
       (= (cals-plato Canelones_de_Carne) 366)
       (= (cals-plato Pastel_de_Verduras) 843)
       (= (cals-plato Fingers_de_Pollo) 601)
       (= (cals-plato Falafel) 450)
       (= (cals-plato Albondigas) 539)
       (= (cals-plato Rape_al_Limon) 473)
```

```
        (= (cals-plato Salmon_Ahumado) 804)
        (= (cals-plato Calsotada) 440)
        (= (cals-plato Costillas_de_Cerdo) 451)
        (= (cals-plato Crema_Zanahoria) 817)
        (= (cals-plato Lasana_de_Atun) 826)
        (= (cals-plato Sopa_de_Miso) 419)
        (= (cals-plato Milanesa_Napolitana) 884)
        (= (cals-plato Bacalao) 805)
        (= (cals-plato Calamares_a_la_Romana) 649)
        (= (cals-plato Raviolli) 429)

    )
    (:goal (
         forall (?d - dia) (servido ?d)
        )
    )
)
```

Resultado:

step 0: PREPARAR_MENU LUNES GAZPACHO SOPA_DE_MISO SOPA SOPA
1: PREPARAR_MENU MARTES MACARRONES_BOLONESA CALSOTADA PASTA VEGETAL
2: PREPARAR_MENU MIERCOLES FIDEUA_CON_SEPIA RAVIOLLI PESCADO PASTA
3: PREPARAR_MENU JUEVES CANELONES_DE_CARNE MILANESA_NAPOLITANA CARNE CARNE
4: PREPARAR_MENU VIERNES PATATAS_CALIU CALAMARES_A_LA_ROMANA VEGETAL PESCADO

➢ 2do Juego de Prueba

```
(define (problem catering-16platos-9incmpls)
    (:domain catering)
    (:objects Cuscus Canelones_de_Carne Potaje_de_Alubias Croquetas Gazpacho
Judias_Verdes Risotto_de_Pescado Paella Sushi Ensalada_de_Col Escudella Ramen
Huevos_Rellenos Macarrones_Bolonesa Pastel_de_Pan Crema_Calabacin - primero
              Canelones_de_Pollo Raviolli Costillas_de_Cerdo
Calamares_a_la_Romana Milanesa_Napolitana Tortilla_de_Espinacas
Fingers_de_Pollo Suquet_de_Pescado Crema_Zanahoria Lasana_de_Atun
Rape_al_Limon Salmon_Ahumado Albondigas Calsotada Sopa_de_Miso
Merluza_a_Naranja - segundo
              Pescado Carne Pasta Vegetal Sopa - tipo
              Viernes Jueves Miercoles Martes Lunes - dia
    )
    (:init
       (= (cals-plato Cuscus) 350)
       (= (cals-plato Canelones_de_Carne) 460)
       (= (cals-plato Potaje_de_Alubias) 710)
       (= (cals-plato Croquetas) 491)
       (= (cals-plato Gazpacho) 354)
       (= (cals-plato Judias_Verdes) 366)
       (= (cals-plato Risotto_de_Pescado) 648)
```

```
        (= (cals-plato Paella) 253)
        (= (cals-plato Sushi) 488)
        (= (cals-plato Ensalada_de_Col) 558)
        (= (cals-plato Escudella) 613)
        (= (cals-plato Ramen) 578)
        (= (cals-plato Huevos_Rellenos) 251)
        (= (cals-plato Macarrones_Bolonesa) 363)
        (= (cals-plato Pastel_de_Pan) 350)
        (= (cals-plato Crema_Calabacin) 576)
        (= (cals-plato Canelones_de_Pollo) 830)
        (= (cals-plato Raviolli) 527)
        (= (cals-plato Costillas_de_Cerdo) 797)
        (= (cals-plato Calamares_a_la_Romana) 670)
        (= (cals-plato Milanesa_Napolitana) 720)
        (= (cals-plato Tortilla_de_Espinacas) 405)
        (= (cals-plato Fingers_de_Pollo) 895)
        (= (cals-plato Suquet_de_Pescado) 778)
        (= (cals-plato Crema_Zanahoria) 505)
        (= (cals-plato Lasana_de_Atun) 617)
        (= (cals-plato Rape_al_Limon) 428)
        (= (cals-plato Salmon_Ahumado) 734)
        (= (cals-plato Albondigas) 663)
        (= (cals-plato Calsotada) 432)
        (= (cals-plato Sopa_de_Miso) 481)
        (= (cals-plato Merluza_a_Naranja) 783)
    )
    (:goal (
         forall (?d - dia) (servido ?d)
        )
    )
)
```

Resultado:

step 0: PREPARAR_MENU VIERNES CREMA_CALABACIN SOPA_DE_MISO VEGETAL SOPA
1: PREPARAR_MENU JUEVES RAMEN CALSOTADA SOPA VEGETAL
2: PREPARAR_MENU MIERCOLES SUSHI RAVIOLLI PESCADO PASTA
3: PREPARAR_MENU MARTES MACARRONES_BOLONESA ALBONDIGAS PASTA CARNE
4: PREPARAR_MENU LUNES HUEVOS_RELLENOS MERLUZA_A_NARANJA CARNE PESCADO

6.6 Extension 5

En está última extensión añadiremos el precio del plato y la métrica para minimizar el coste total del menú semanal. Seguiremos la tónica del último apartado y utilizaremos los anteriores juegos de prueba, además mostraremos la única parte del problema que se modifica y asumimos que lo demás es exactamente igual que en los apartados anteriores

➢ 1er Juego de Prueba

```
(define (problem catering-15platos-3incmpls)
    (:domain catering)
    (:objects Pastel_de_Pan Spaghettis_Bolonesa Risotto_de_Pescado
Potaje_de_Alubias Sushi Crema_Calabacin Cuscus Patatas_Caliu
Tagliatelle_Bolonesa Croquetas Gazpacho Fideua_con_Sepia Macarrones_Bolonesa
Judias_Verdes Canelones_de_Carne - primero
              Pastel_de_Verduras Fingers_de_Pollo Falafel Albondigas
Rape_al_Limon Salmon_Ahumado Calsotada Costillas_de_Cerdo Crema_Zanahoria
Lasana_de_Atun Sopa_de_Miso Milanesa_Napolitana Bacalao Calamares_a_la_Romana
Raviolli - segundo
              Pescado Carne Pasta Vegetal Sopa - tipo
              Lunes Martes Miercoles Jueves Viernes - dia
    )
    (:init
      ..
       (= (precio-plato Pastel_de_Pan) 5)
       (= (cals-plato Pastel_de_Pan) 348)
       (= (precio-plato Spaghettis_Bolonesa) 7)
       (= (cals-plato Spaghettis_Bolonesa) 352)
       (= (precio-plato Risotto_de_Pescado) 7)
       (= (cals-plato Risotto_de_Pescado) 330)
       (= (precio-plato Potaje_de_Alubias) 8)
       (= (cals-plato Potaje_de_Alubias) 413)
       (= (precio-plato Sushi) 6)
       (= (cals-plato Sushi) 725)
       (= (precio-plato Crema_Calabacin) 9)
       (= (cals-plato Crema_Calabacin) 445)
       (= (precio-plato Cuscus) 7)
       (= (cals-plato Cuscus) 582)
       (= (precio-plato Patatas_Caliu) 8)
       (= (cals-plato Patatas_Caliu) 613)
       (= (precio-plato Tagliatelle_Bolonesa) 5)
       (= (cals-plato Tagliatelle_Bolonesa) 393)
       (= (precio-plato Croquetas) 5)
       (= (cals-plato Croquetas) 410)
       (= (precio-plato Gazpacho) 7)
       (= (cals-plato Gazpacho) 693)
       (= (precio-plato Fideua_con_Sepia) 9)
       (= (cals-plato Fideua_con_Sepia) 574)
```

```
        (= (precio-plato Macarrones_Bolonesa) 6)
        (= (cals-plato Macarrones_Bolonesa) 580)
        (= (precio-plato Judias_Verdes) 9)
        (= (cals-plato Judias_Verdes) 272)
        (= (precio-plato Canelones_de_Carne) 5)
        (= (cals-plato Canelones_de_Carne) 366)
        (= (precio-plato Pastel_de_Verduras) 13)
        (= (cals-plato Pastel_de_Verduras) 843)
        (= (precio-plato Fingers_de_Pollo) 9)
        (= (cals-plato Fingers_de_Pollo) 601)
        (= (precio-plato Falafel) 10)
        (= (cals-plato Falafel) 450)
        (= (precio-plato Albondigas) 12)
        (= (cals-plato Albondigas) 539)
        (= (precio-plato Rape_al_Limon) 14)
        (= (cals-plato Rape_al_Limon) 473)
        (= (precio-plato Salmon_Ahumado) 10)
        (= (cals-plato Salmon_Ahumado) 804)
        (= (precio-plato Calsotada) 10)
        (= (cals-plato Calsotada) 440)
        (= (precio-plato Costillas_de_Cerdo) 8)
        (= (cals-plato Costillas_de_Cerdo) 451)
        (= (precio-plato Crema_Zanahoria) 14)
        (= (cals-plato Crema_Zanahoria) 817)
        (= (precio-plato Lasana_de_Atun) 9)
        (= (cals-plato Lasana_de_Atun) 826)
        (= (precio-plato Sopa_de_Miso) 12)
        (= (cals-plato Sopa_de_Miso) 419)
        (= (precio-plato Milanesa_Napolitana) 11)
        (= (cals-plato Milanesa_Napolitana) 884)
        (= (precio-plato Bacalao) 13)
        (= (cals-plato Bacalao) 805)
        (= (precio-plato Calamares_a_la_Romana) 8)
        (= (cals-plato Calamares_a_la_Romana) 649)
        (= (precio-plato Raviolli) 14)
        (= (cals-plato Raviolli) 429)
        (= (total-cost) 0)
    )
    (:goal (
         forall (?d - dia) (servido ?d)
        )
    )
    (:metric minimize (total-cost))
)
```

Resultado:

step 0: PREPARAR_MENU LUNES GAZPACHO CALSOTADA SOPA VEGETAL
1: PREPARAR_MENU MARTES CANELONES_DE_CARNE CALAMARES_A_LA_ROMANA CARNE PESCADO
2: PREPARAR_MENU MIERCOLES CUSCUS SOPA_DE_MISO VEGETAL SOPA
3: PREPARAR_MENU JUEVES MACARRONES_BOLONESA RAVIOLLI PASTA PASTA
4: PREPARAR_MENU VIERNES SUSHI COSTILLAS_DE_CERDO PESCADO CARNE

➢ 2do Juego de Prueba

```
(define (problem catering-16platos-9incmpls)
    (:domain catering)
    (:objects Cuscus Canelones_de_Carne Potaje_de_Alubias Croquetas Gazpacho
Judias_Verdes Risotto_de_Pescado Paella Sushi Ensalada_de_Col Escudella Ramen
Huevos_Rellenos Macarrones_Bolonesa Pastel_de_Pan Crema_Calabacin - primero
              Canelones_de_Pollo Raviolli Costillas_de_Cerdo
Calamares_a_la_Romana Milanesa_Napolitana Tortilla_de_Espinacas
Fingers_de_Pollo Suquet_de_Pescado Crema_Zanahoria Lasana_de_Atun
Rape_al_Limon Salmon_Ahumado Albondigas Calsotada Sopa_de_Miso
Merluza_a_Naranja - segundo
              Pescado Carne Pasta Vegetal Sopa - tipo
              Viernes Jueves Miercoles Martes Lunes - dia
    )
    (:init
      ..
       (= (precio-plato Cuscus) 6)
       (= (cals-plato Cuscus) 350)
       (= (precio-plato Canelones_de_Carne) 9)
       (= (cals-plato Canelones_de_Carne) 460)
       (= (precio-plato Potaje_de_Alubias) 9)
       (= (cals-plato Potaje_de_Alubias) 710)
       (= (precio-plato Croquetas) 5)
       (= (cals-plato Croquetas) 491)
       (= (precio-plato Gazpacho) 8)
       (= (cals-plato Gazpacho) 354)
       (= (precio-plato Judias_Verdes) 9)
       (= (cals-plato Judias_Verdes) 366)
       (= (precio-plato Risotto_de_Pescado) 7)
       (= (cals-plato Risotto_de_Pescado) 648)
       (= (precio-plato Paella) 5)
       (= (cals-plato Paella) 253)
       (= (precio-plato Sushi) 7)
       (= (cals-plato Sushi) 488)
       (= (precio-plato Ensalada_de_Col) 8)
       (= (cals-plato Ensalada_de_Col) 558)
       (= (precio-plato Escudella) 8)
       (= (cals-plato Escudella) 613)
       (= (precio-plato Ramen) 9)
       (= (cals-plato Ramen) 578)
       (= (precio-plato Huevos_Rellenos) 8)
       (= (cals-plato Huevos_Rellenos) 251)
       (= (precio-plato Macarrones_Bolonesa) 5)
       (= (cals-plato Macarrones_Bolonesa) 363)
       (= (precio-plato Pastel_de_Pan) 9)
       (= (cals-plato Pastel_de_Pan) 350)
       (= (precio-plato Crema_Calabacin) 9)
       (= (cals-plato Crema_Calabacin) 576)
       (= (precio-plato Canelones_de_Pollo) 12)
       (= (cals-plato Canelones_de_Pollo) 830)
```

```
        (= (precio-plato Raviolli) 10)
        (= (cals-plato Raviolli) 527)
        (= (precio-plato Costillas_de_Cerdo) 8)
        (= (cals-plato Costillas_de_Cerdo) 797)
        (= (precio-plato Calamares_a_la_Romana) 12)
        (= (cals-plato Calamares_a_la_Romana) 670)
        (= (precio-plato Milanesa_Napolitana) 10)
        (= (cals-plato Milanesa_Napolitana) 720)
        (= (precio-plato Tortilla_de_Espinacas) 12)
        (= (cals-plato Tortilla_de_Espinacas) 405)
        (= (precio-plato Fingers_de_Pollo) 12)
        (= (cals-plato Fingers_de_Pollo) 895)
        (= (precio-plato Suquet_de_Pescado) 13)
        (= (cals-plato Suquet_de_Pescado) 778)
        (= (precio-plato Crema_Zanahoria) 14)
        (= (cals-plato Crema_Zanahoria) 505)
        (= (precio-plato Lasana_de_Atun) 10)
        (= (cals-plato Lasana_de_Atun) 617)
        (= (precio-plato Rape_al_Limon) 10)
        (= (cals-plato Rape_al_Limon) 428)
        (= (precio-plato Salmon_Ahumado) 11)
        (= (cals-plato Salmon_Ahumado) 734)
        (= (precio-plato Albondigas) 14)
        (= (cals-plato Albondigas) 663)
        (= (precio-plato Calsotada) 13)
        (= (cals-plato Calsotada) 432)
        (= (precio-plato Sopa_de_Miso) 12)
        (= (cals-plato Sopa_de_Miso) 481)
        (= (precio-plato Merluza_a_Naranja) 13)
        (= (cals-plato Merluza_a_Naranja) 783)
        (= (total-cost) 0)
    )
    (:goal (
         forall (?d - dia) (servido ?d)
        )
    )
    (:metric minimize (total-cost))
)
```

Resultado:

step 0: PREPARAR_MENU VIERNES ESCUDELLA COSTILLAS_DE_CERDO SOPA CARNE
1: PREPARAR_MENU JUEVES MACARRONES_BOLONESA SALMON_AHUMADO PASTA PESCADO
2: PREPARAR_MENU MIERCOLES PAELLA SUQUET_DE_PESCADO PESCADO SOPA
3: PREPARAR_MENU MARTES CREMA_CALABACIN CALSOTADA VEGETAL VEGETAL
4: PREPARAR_MENU LUNES CROQUETAS RAVIOLLI CARNE PASTA

7. Conclusiones

En esta práctica hemos resuelto paso a paso un problema de planificación automática, empezando desde el nivel básico de resolución hasta la quinta extensión.

Aun teniendo menos tiempo que en los anteriores laboratorios, hemos podido ver los tipos de problemas y que técnicas de planificación aplicar en cada momento, además de aprender de estas y explotar su potencial así como poner en práctica los conocimientos adquiridos durante la última parte de la asignatura.

Hemos trabajado con lenguajes formales de definición de dominios como el PDDL y hemos usado el potencial del Fast Forward y sus optimizaciones. Además tenemos implementado un generador de problemas en *Python*.

Hemos modelado el problema siguiendo la guia del enunciado y de lo aprendido en las clases de teoría, y hemos acabado viendo que este sistema es capaz de ofrecer soluciones correctas en un tiempo más que aceptable, ofreciendo menús variados con restricciones muy concretas.