DOCUMENTACIÓN PRÁCTICA DE PLANIFICACIÓN

INTELIGENCIA ARTIFICIAL 2024-2025 Q1

GRADO EN INGENIERÍA INFORMÁTICA - UPC FIB

DAVID MORAIS, JOAN VILA, NURIA ENSEÑAT

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Facultat d'Informàtica de Barcelona

ÍNDICE

1. INTRODUCCIÓN

REDFLIX, una empresa que distribuye contenidos audiovisuales, nos ha pedido una aplicación capaz de planificar el visionado de las series preferidas de sus usuarios, haciendo un plan de visionado diario que le recomiende cuándo ver las películas o capítulos balanceando el número de contenidos o minutos que ha de ver cada día.

Hay que tener en cuenta varios aspectos a la hora de desarrollar esta herramienta: por un lado, tendremos las películas y los capítulos de series televisivas que el usuario ya ha visto, por otro lado, tendremos aquellos contenidos que planea ver. Hay dos tipos de contenidos disponibles: los contenidos predecesores, que se han de ver antes de otro contenido, y los contenidos paralelos, que es recomendable verlos a la vez.

Veremos que hay varios niveles de resolución del problema que se comentarán en las siguientes secciones.

2. MODELADO DOMINIO

Tipos

Para los tipos simplemente tenemos `content` que representa un contenido y `day` que representan los días.

Predicados

- *`watch ?cont`*: Representa que un contenido ya se ha visto.
- *`toWatch ?cont`*: Representa un contenido que se quiere ver.
- *`predecessor ?cont1 ?cont2`*: Representa que el contenido 1 es predecesor del contenido 2. Quiere decir que tendremos que ver antes el contenido 1 que el 2.
- *`previous ?d1 ?d2`*: Representa que el día 1 es anterior al día 2. Por ejemplo tenemos el d0, d1 y d2, entonces d0 es previous a d1 y d2.
- *`watchedDay ?cont ?d`*: Representa en qué día hemos visto el contenido. Nos es útil para controlar el día que hemos visto el contenido tanto para paralelos como predecesores
- `before ?d1 ?d2`: Representa que d1 es el día anterior a d2, cada día solo puede tener un before, por ejemplo si tenemos 3 días sería, d0-d1, d1-d2.
- `parallel ?cont1 ?cont2`: Representa que el contenido cont1 es paralelo a cont2, eso quiere decir que ambos se tienen que ver en un dia de diferencia como máximo.

3. OPERADORES

Funciones

- `contentsDay ?d`: Representa la cantidad de contenidos que hemos visto en el día d.
- `minutesWatched ?d`: Representa cuántos minutos hemos visto en el día d.
- `duration ?cont`: Representa la duración de cada contenido en minutos.

Acciones

- `addPredecessors:` Se comprueba que cont1 sea predecesor de cont2, que tengamos como pendiente para ver cont2 y que este no haya sido visto. En este caso simplemente añadimos a los pendientes cont1 para poder ver cont2.
- `addParallels`: Es parecido al anterior, primero comprobamos que cont1 es paralelo a cont2 o viceversa, que no tengamos como pendiente uno de los dos y no lo hayamos visto. El efecto es que simplemente ponemos como pendiente el contenido paralelo que no hayamos visto, sea cont1 o cont2.
- `watch:` Esta es la acción más compleja de las 3. Primero comprobamos que tengamos el contenido pendiente de ver y que aún no esté visto. Si es el caso, tiene que cumplir: que para contenido predecesor debe ya estar visto y se haya visto en un día anterior al actual. Para todo contenido paralelo, si están pendientes de ver se ven en el mismo día o en el día anterior. Si todo esto se cumple, marcamos como visto el contenido y registramos en qué día se ha visto.

  Para la extensión 3, simplemente añadimos a la precondición que el número de contenidos vistos ese día sea menor de 3 y incrementamos el contador en el efecto.

  Para la extensión 4 es muy parecido, solo que tenemos que comprobar que la suma de los minutos que llevamos en el día más lo que dura el contenido no supere los 200 minutos y en el efecto sumamos los minutos del contenido a los minutos consumidos en el día.

4. DESARROLLO DE LOS NIVELES

4.1 Nivel básico

En el nivel básico encontramos que los contenidos tienen de 0 a 1 predecesores pero no tienen ningún contenido paralelo.

En el archivo del dominio tendremos en cuenta que si se quiere ver un contenido que tiene predecesor, éste se tendrá que ver antes del contenido deseado. Para ello, tenemos una acción `addPredecessors` que gestiona esta restricción añadiendo el contenido predecesor de todos los contenidos que se quieren ver a `toWatch`, si existe. También, tendremos una acción llamada `watch`, que marcará los contenidos como vistos (de `toWatch` a `watch`).

En el archivo del problema hemos definido únicamente un objeto, `content` y dos predicados: `toWatch` y `predecessor`. El objetivo será que se hayan visto todos los contenidos `toWatch`.

4.2 Nivel 1

En el nivel 1, los contenidos tienen de 0 a N predecesores, pero no tienen contenidos paralelos. El hecho de que existan dependencias entre los contenidos implica que el usuario tendrá que ver anteriormente aquellos contenidos que son predecesores de los que quiere ver en su plan de visionado.

En el archivo del dominio utilizado para el nivel anterior añadiremos los predicados `previous`, `before` y `watchedDay`, de esta manera podremos controlar en la acción `watch` que se hayan visto previamente todos los contenidos predecesores.

Por este motivo, modificaremos el archivo del problema de este nivel añadiendo el objeto `day`, y el predicado `previous`. El objetivo es que se hayan visto todos los contenidos `toWatch`.

4.3 Nivel 2

En el nivel 2, los contenidos tienen de 0 a N predecesores y de 0 a M contenidos paralelos. Esto implica que el usuario además de tener que ver anteriormente los contenidos predecesores a aquellos que quiere ver, tendrá que ver todos los contenidos paralelos que pertenecen al plan de visionado el día anterior o el mismo día.

En el archivo del dominio utilizado para el nivel anterior añadimos los predicados `parallel` y `before`, que nos ayudará a gestionar la visualización de los contenidos paralelos, que se tienen que ver el día anterior o el mismo día al contenido deseado. Además, añadiremos la acción `addParallels` y en la acción `watch` añadiremos la lógica necesaria para asegurarnos de que cumplimos la restricción de los contenidos paralelos.

En el archivo del problema tendremos que añadir la definición de los predicados `before` y `parallel`, y el objetivo será que se hayan visto todos los contenidos `toWatch`.

4.4 Nivel 3

En la nivel 3 controlaremos que no se coloquen más de 3 contenidos al día.

Para esto, modificaremos el archivo del dominio utilizado para el nivel 2 añadiendo fluentes para así poder declarar la función `contentsDay`. Para controlar la restricción de no ver más de 3 contenidos al día miraremos que `contentsDay` sea menor a 3, y cada vez que añadamos la visualización de un contenido en un día, aumentaremos `contentsDay` en uno.

En el archivo del problema añadiremos los siete `contentsDay` inicializados a 0.

4.5 Nivel 4

En el nivel 4 los contenidos tendrán asignados el número de minutos de duración, por lo que tendremos que controlar que en el plan generado no se superen los 200 minutos al día.

Para este último nivel partiremos de los archivos utilizados para el nivel 2. En cuanto al archivo del dominio añadiremos el uso de fluentes y así declarar dos funciones: `minutesWatched` y `duration`. En la acción `watch` controlaremos que los `minutesWatched` sean menor o igual a 200 y le iremos sumando la duración del contenido (`duration`) cada vez que lo marquemos como visto.

En el archivo del problema tendremos que añadir la duración en minutos de los contenidos y los minutos vistos para cada día inicializados a 0.

5. JUEGOS DE PRUEBA

5.1 Nivel 1

Problema 1

Estado inicial:

```
(:init
    (previous d0 d1) (previous d0 d2) (previous d0 d3) (previous d0 d4)
    (previous d1 d2) (previous d1 d3) (previous d1 d4) (previous d2 d3)
    (previous d2 d4) (previous d3 d4)

    (predecessor S0_0 S0_1) (predecessor S0_1 S0_2)
     (predecessor S1_0 S1_1) (predecessor S1_1 S1_2) (predecessor S1_2
S1_3)
    (predecessor S2_0 S2_1) (predecessor S2_1 S2_2)

    (toWatch S0_2) (toWatch S1_3) (toWatch S2_2) (toWatch M0)
    (toWatch M1) (toWatch M2) (toWatch M3)
)
```

En el estado inicial se definen las relaciones de precedencia entre días mediante `previous`, asegurando así que el plan respete un orden cronológico. Además, se especifican relaciones de predecesor entre los episodios de las series, como `S0_0` → `S0_1` → `S0_2`, entre otras. Por último, se introducen los contenidos pendientes de visualización (`toWatch`), se incluyen los episodios finales de las series y las películas `M0`, `M1`, `M2` y `M3`.

Plan encontrado:

```
step    0: WATCH M3 D0
        1: WATCH M2 D0
        2: WATCH M1 D0
        3: WATCH M0 D0
        4: ADDPREDECESSORS S2_1 S2_2
        5: ADDPREDECESSORS S2_0 S2_1
        6: WATCH S2_0 D0
        7: WATCH S2_1 D1
        8: WATCH S2_2 D2
        9: ADDPREDECESSORS S1_2 S1_3
       10: ADDPREDECESSORS S1_1 S1_2
       11: ADDPREDECESSORS S1_0 S1_1
       12: WATCH S1_0 D0
       13: WATCH S1_1 D1
       14: WATCH S1_2 D2
       15: WATCH S1_3 D3
       16: ADDPREDECESSORS S0_1 S0_2
       17: ADDPREDECESSORS S0_0 S0_1
       18: WATCH S0_0 D0
       19: WATCH S0_1 D1
       20: WATCH S0_2 D2
       21: REACH-GOAL
```

Como se ve el planificador encuentra un plan comenzando con la visualización de todas las películas en el día 0. Y luego procesa las relaciones de precedencia en las series y se organizan de manera que consigue que se visualicen de forma correcta.. El plan respeta todas las restricciones de precedencia, completándose en 22 pasos.

Problema 2

Estado inicial:

```
(:init
    (previous d0 d1) (previous d0 d2) (previous d0 d3) (previous d0 d4)
    (previous d1 d2) (previous d1 d3) (previous d1 d4) (previous d2 d3)
    (previous d2 d4) (previous d3 d4)

    (predecessor S0_0 S0_1) (predecessor S0_1 S0_2)
    (predecessor S1_0 S1_1) (predecessor S1_1 S1_2)

    (toWatch S0_2) (toWatch S1_2) (toWatch M0)
    (toWatch M1) (toWatch M2)
)
```

En este problema, el estado inicial también establece un orden temporal entre los días mediante `previous` y relaciones de predecesor entre los episodios. Y posteriormente se introducen los contenidos pendientes (`toWatch`).

Plan encontrado:

```
step    0: WATCH M2 D0
        1: WATCH M1 D0
        2: WATCH M0 D0
        3: ADDPREDECESSORS S1_1 S1_2
        4: ADDPREDECESSORS S1_0 S1_1
        5: WATCH S1_0 D0
        6: WATCH S1_1 D1
        7: WATCH S1_2 D2
        8: ADDPREDECESSORS S0_1 S0_2
        9: ADDPREDECESSORS S0_0 S0_1
       10: WATCH S0_0 D0
       11: WATCH S0_1 D1
       12: WATCH S0_2 D2
       13: REACH-GOAL
```

El planificador genera un plan donde inicialmente se visualizan las películas necesarias en el día 0. Posteriormente, se procesan las series: los episodios se distribuyen entre los días para poder seguir el mismo esquema temporal. El plan logra completar la meta en 14 pasos, respetando las relaciones de precedencia y asegurando que todos los contenidos marcados como `toWatch` sean visualizados.

5.2 Nivel 2

Problema 1

Estado inicial:

```
(:init
    (previous d0 d1)
    (previous d0 d2)
    (previous d0 d3)
    (previous d1 d2)
    (previous d1 d3)
    (previous d2 d3)

    (predecessor S0_0 S0_1)
    (predecessor S0_1 S0_2)
    (predecessor S1_0 S1_1)
    (predecessor S1_1 S1_2)
    (predecessor S2_0 S2_1)
    (predecessor S2_1 S2_2)
    (predecessor S3_0 S3_1)
    (predecessor S3_1 S3_2)

    (toWatch S0_2)
    (toWatch S1_2)
    (toWatch S2_2)
    (toWatch S3_2)
    (toWatch M0)
    (toWatch M1)
    (toWatch M2)

    (before d0 d1)
    (before d1 d2)
    (before d2 d3)
```

```
    (parallel M2 S1_1)
    (parallel S0_1 S0_0)
    (parallel S2_0 S3_0)
)
```

En este problema, el estado inicial establece un orden cronológico entre los días mediante las relaciones `previous` y `before`. Las series también tienen relaciones de precedencia entre episodios, como `S0_0` precede a `S0_1` y `S0_1` precede a `S0_2`. Ahora también se incluyen definiciones de paralelismo, como `M2` paralelo a `S1_1`, `S0_1`. Los contenidos marcados como `toWatch` incluyen los episodios finales de las series y todas las películas.

Plan encontrado:

```
step    0: WATCH M1 D0
        1: WATCH M0 D0
        2: ADDPREDECESSORS S3_1 S3_2
        3: ADDPREDECESSORS S3_0 S3_1
        4: ADDPARALLELS S2_0 S3_0
        5: WATCH S2_0 D0
        6: WATCH S3_0 D0
        7: WATCH S3_1 D1
        8: WATCH S3_2 D2
        9: ADDPREDECESSORS S2_1 S2_2
       10: WATCH S2_1 D1
       11: WATCH S2_2 D2
       12: ADDPREDECESSORS S1_1 S1_2
       13: ADDPREDECESSORS S1_0 S1_1
       14: WATCH S1_0 D0
       15: WATCH S1_1 D1
       16: WATCH S1_2 D2
       17: WATCH M2 D1
       18: ADDPREDECESSORS S0_1 S0_2
       19: ADDPARALLELS S0_1 S0_0
       20: WATCH S0_0 D0
       21: WATCH S0_1 D1
```

```
22: WATCH S0_2 D2
23: REACH-GOAL
```

El planificador genera un plan que comienza con la visualización de las películas en el día 0, posteriormente se ejecutan las acciones `ADDPREDECESSORS` para procesar las relaciones de precedencia en las series. Finalmente se distribuyen los episodios tanto en paralelo como en los días dichos. El plan respeta todas las restricciones, completándose en 24 pasos.

Problema 2

Estado inicial:

```
(:init
    (previous d0 d1)
    (previous d0 d2)
    (previous d0 d3)
    (previous d0 d4)
    (previous d0 d5)
    (previous d1 d2)
    (previous d1 d3)
    (previous d1 d4)
    (previous d1 d5)
    (previous d2 d3)
    (previous d2 d4)
    (previous d2 d5)
    (previous d3 d4)
    (previous d3 d5)
    (previous d4 d5)

    (predecessor S0_0 S0_1)
    (predecessor S1_0 S1_1)
    (predecessor S2_0 S2_1)
    (predecessor S3_0 S3_1)
    (predecessor S3_1 S3_2)

    (toWatch S0_1)
    (toWatch S1_1)
    (toWatch S2_1)
    (toWatch S3_2)
    (toWatch M0)
    (toWatch M1)
    (toWatch M2)
```

```
    (toWatch M3)

    (before d0 d1)
    (before d1 d2)
    (before d2 d3)
    (before d3 d4)
    (before d4 d5)

    (parallel S3_2 S2_1)
    (parallel M3 S0_1)
    (parallel M3 S3_1)
)
```

El estado inicial de este problema también define un orden temporal entre los días mediante las relaciones `previous` y `before`. Las series tienen relaciones de precedencia, como anteriormente hemos explicado. También se introducen qué contenidos son paralelos incluyen como `S3_2` paralelo a `S2_1` y `M3` paralelo a `S0_1` y `S3_1`. Los contenidos pendientes de visualización (`toWatch`) son `S0_1`, `S1_1`, `S2_1`, `S3_2`, `M0`, `M1`, `M2`, y `M3`.

Plan encontrado:

```
step    0: WATCH M2 D0
        1: WATCH M1 D0
        2: WATCH M0 D0
        3: ADDPREDECESSORS S3_1 S3_2
        4: ADDPREDECESSORS S3_0 S3_1
        5: WATCH S3_0 D0
        6: WATCH S3_1 D1
        7: WATCH S3_2 D2
        8: WATCH M3 D1
        9: ADDPREDECESSORS S2_0 S2_1
       10: WATCH S2_0 D0
       11: WATCH S2_1 D2
       12: ADDPREDECESSORS S1_0 S1_1
       13: WATCH S1_0 D0
```

```
14: WATCH S1_1 D1
15: ADDPREDECESSORS S0_0 S0_1
16: WATCH S0_0 D0
17: WATCH S0_1 D1
18: REACH-GOAL
```

El planificador genera un plan en el que primero se visualizan las películas en el día 0. Posteriormente, se procesan las relaciones de precedencia en las series. Paralelamente, se procesan los episodios, que se distribuyen entre los días. El plan respeta todas las restricciones de precedencia, paralelismo y tiempo, completándose en 19 pasos.

5.3 Nivel 3

Problema 1

Estado inicial:

```
(:init
    (before d0 d1)
    (before d1 d2)
    (before d2 d3)
    (before d3 d4)
    (before d4 d5)
    (before d5 d6)

    (previous d0 d1)
    (previous d0 d2)
    (previous d0 d3)
    (previous d0 d4)
    (previous d0 d5)
    (previous d0 d6)
    (previous d1 d2)
    (previous d1 d3)
    (previous d1 d4)
    (previous d1 d5)
    (previous d1 d6)
    (previous d2 d3)
    (previous d2 d4)
    (previous d2 d5)
    (previous d2 d6)
    (previous d3 d4)
    (previous d3 d5)
    (previous d3 d6)
    (previous d4 d5)
    (previous d4 d6)
    (previous d5 d6)
```

```
    (predecessor S0_0 S0_1)
    (predecessor S0_1 S0_2)
    (predecessor S1_0 S1_1)
    (predecessor S2_0 S2_1)
    (predecessor S2_1 S2_2)
    (predecessor S3_0 S3_1)
    (predecessor S3_1 S3_2)
    (predecessor S3_2 S3_3)

    (toWatch S0_2)
    (toWatch S1_1)
    (toWatch S2_2)
    (toWatch S3_3)
    (toWatch M0)
    (toWatch M1)
    (toWatch M2)
    (toWatch M3)
    (toWatch M4)

    (parallel M0 M2)
    (= (contentsDay d0) 0)
    (= (contentsDay d1) 0)
    (= (contentsDay d2) 0)
    (= (contentsDay d3) 0)
    (= (contentsDay d4) 0)
    (= (contentsDay d5) 0)
    (= (contentsDay d6) 0)
)
```

En este problema, el estado inicial establece un orden cronológico estricto entre los días mediante las relaciones `before` y `previous`. Se definen relaciones de precedencia entre episodios de las series. Y se establece el paralelismo entre algunos contenidos. Todos los contenidos comienzan con el estado `toWatch`, incluidos los episodios finales de las series y las películas.

Plan encontrado:

```
step    0: WATCH M0 D0
        1: WATCH M1 D0
        2: WATCH M2 D0
        3: WATCH M3 D1
        4: WATCH M4 D1
        5: ADDPREDECESSORS S3_2 S3_3
        6: ADDPREDECESSORS S3_1 S3_2
        7: ADDPREDECESSORS S3_0 S3_1
        8: WATCH S3_0 D1
        9: WATCH S3_1 D2
       10: WATCH S3_2 D3
       11: WATCH S3_3 D4
       12: ADDPREDECESSORS S2_1 S2_2
       13: ADDPREDECESSORS S2_0 S2_1
       14: WATCH S2_0 D2
       15: WATCH S2_1 D3
       16: WATCH S2_2 D4
       17: ADDPREDECESSORS S1_0 S1_1
       18: WATCH S1_0 D2
       19: WATCH S1_1 D3
       20: ADDPREDECESSORS S0_1 S0_2
       21: ADDPREDECESSORS S0_0 S0_1
       22: WATCH S0_0 D4
       23: WATCH S0_1 D5
       24: WATCH S0_2 D6
       25: REACH-GOAL
```

El plan generado por el planificador comienza con la visualización de las películas `M0`, `M1` y `M2` en el día 0, y `M3` y `M4` en el día 1. Luego, se procesan las relaciones de precedencia en las series: `S3_0`, `S3_1`, `S3_2` y `S3_3` se completan entre los días 1 y 4; `S2_0`, `S2_1` y `S2_2` se distribuyen entre los días 2 y 4. Finalmente, los episodios `S1_0` y `S1_1` se ven en los días 2 y 3, mientras que los episodios de `S0` (`S0_0`, `S0_1`, `S0_2`) se completan entre los días 4 y 6. Este plan cumple con

todas las restricciones de precedencia y paralelismo y respetando que no exista más de 3 contenidos en un mismo día, en 26 pasos.

Problema 2

Estado inicial:

```
(:init
    (before d0 d1)
    (before d1 d2)
    (before d2 d3)
    (before d3 d4)
    (before d4 d5)

    (previous d0 d1)
    (previous d0 d2)
    (previous d0 d3)
    (previous d0 d4)
    (previous d0 d5)
    (previous d1 d2)
    (previous d1 d3)
    (previous d1 d4)
    (previous d1 d5)
    (previous d2 d3)
    (previous d2 d4)
    (previous d2 d5)
    (previous d3 d4)
    (previous d3 d5)
    (previous d4 d5)

    (predecessor S0_0 S0_1)
    (predecessor S0_1 S0_2)
    (predecessor S1_0 S1_1)
    (predecessor S1_1 S1_2)
    (predecessor S2_0 S2_1)
    (predecessor S2_1 S2_2)
    (predecessor S2_2 S2_3)
```

```
    (toWatch S0_2)
    (toWatch S1_2)
    (toWatch S2_3)
    (toWatch M0)
    (toWatch M1)
    (toWatch M2)
    (toWatch M3)
    (toWatch M4)

    (parallel M1 S2_0)
    (parallel M3 S0_1)

    (= (contentsDay d0) 0)
    (= (contentsDay d1) 0)
    (= (contentsDay d2) 0)
    (= (contentsDay d3) 0)
    (= (contentsDay d4) 0)
    (= (contentsDay d5) 0)
)
```

El estado inicial establece nuevamente todo lo anterior con ligeros cambios. Los contenidos pendientes de visualización (`toWatch`) incluyen los episodios finales de las series (`S0_2`, `S1_2`, `S2_3`) y las películas (`M0`, `M1`, `M2`, `M3`, `M4`).

Plan encontrado:

```
step    0: WATCH M0 D0
        1: WATCH M2 D0
        2: WATCH M4 D0
        3: ADDPARALLELS M1 S2_0
        4: WATCH S2_0 D1
        5: WATCH M1 D1
        6: ADDPREDECESSORS S2_2 S2_3
        7: ADDPREDECESSORS S2_1 S2_2
        8: WATCH S2_1 D2
        9: WATCH S2_2 D3
       10: WATCH S2_3 D4
       11: ADDPREDECESSORS S1_1 S1_2
       12: ADDPREDECESSORS S1_0 S1_1
       13: WATCH S1_0 D2
       14: WATCH S1_1 D3
       15: WATCH S1_2 D4
       16: ADDPREDECESSORS S0_1 S0_2
       17: WATCH M3 D3
       18: ADDPREDECESSORS S0_0 S0_1
       19: WATCH S0_0 D2
       20: WATCH S0_1 D4
       21: WATCH S0_2 D5
       22: REACH-GOAL
```

El planificador encuentra una solución en la que primero se visualizan las películas `M0`, `M2` y `M4` en el día 0, mientras que `M1` y `S2_0` se procesan en el día 1. Después, se introducen los episodios de `S2` entre los días 2 y 4. Paralelamente, los episodios de `S1` se distribuyen entre los días 2 y 4. Finalmente, los episodios de `S0` se completan entre los días 2 y 5, mientras que `M3` se visualiza en el día 3. Este plan cumple con todas las restricciones de precedencia, paralelismo y respeta la condición de que no exista más de 3 contenidos en un mismo día en 23 pasos.

5.4 Nivel 4

Problema 1

Estado inicial:

```
(:init
    (before d0 d1)
    (before d1 d2)
    (before d2 d3)
    (before d3 d4)
    (before d4 d5)

    (previous d0 d1)
    (previous d0 d2)
    (previous d0 d3)
    (previous d0 d4)
    (previous d0 d5)
    (previous d1 d2)
    (previous d1 d3)
    (previous d1 d4)
    (previous d1 d5)
    (previous d2 d3)
    (previous d2 d4)
    (previous d2 d5)
    (previous d3 d4)
    (previous d3 d5)
    (previous d4 d5)

    (predecessor S0_0 S0_1)
    (predecessor S1_0 S1_1)
    (predecessor S2_0 S2_1)
    (predecessor S3_0 S3_1)

    (toWatch S0_1)
```

```
    (toWatch S1_1)
    (toWatch S2_1)
    (toWatch S3_1)
    (toWatch M0)
    (toWatch M1)

    (parallel S1_0 M1)
    (parallel S1_1 S2_0)
    (parallel S3_1 S1_1)

    (= (duration S0_0) 57)
    (= (duration S0_1) 40)
    (= (duration S1_0) 29)
    (= (duration S1_1) 39)
    (= (duration S2_0) 36)
    (= (duration S2_1) 43)
    (= (duration S3_0) 29)
    (= (duration S3_1) 23)
    (= (duration M0) 91)
    (= (duration M1) 90)

    (= (minutesWatched d0) 0)
    (= (minutesWatched d1) 0)
    (= (minutesWatched d2) 0)
    (= (minutesWatched d3) 0)
    (= (minutesWatched d4) 0)
    (= (minutesWatched d5) 0)
)
```

En este problema, el estado inicial establece todo lo explicado anteriormente un orden cronológico entre los días. Se introducen relaciones de precedencia definidas entre las series. Y además, se añaden las relaciones de paralelismo. Ahora se introduce a los contenidos duraciones que limitan el tiempo disponible por día. Los contenidos pendientes (`toWatch`) incluyen los episodios finales de las series y las películas.

Plan encontrado:

```
step    0: WATCH M0 D0
        1: ADDPREDECESSORS S3_0 S3_1
        2: WATCH S3_0 D0
        3: WATCH S3_1 D1
        4: ADDPREDECESSORS S1_0 S1_1
        5: WATCH S1_0 D0
        6: WATCH M1 D1
        7: ADDPARALLELS S1_1 S2_0
        8: WATCH S2_0 D0
        9: WATCH S1_1 D1
       10: WATCH S2_1 D2
       11: ADDPREDECESSORS S0_0 S0_1
       12: WATCH S0_0 D2
       13: WATCH S0_1 D3
       14: REACH-GOAL
```

El planificador en este caso genera un plan en el que primero se visualizan `M0`, `S3_0` y `S1_0` en el día 0, respetando el paralelismo entre ellos. Al día siguiente, se visualizan `S3_1`, `M1` y `S2_0`, los episodios `S2_1` y `S0_0` se completan en el día 2, y finalmente, `S0_1` se ve en el día 3. El plan respeta todas las restricciones de precedencia, paralelismo y duración, completándose en 15 pasos.

Problema 2

Estado inicial:

```
(:init
    (before d0 d1)
    (before d1 d2)
    (before d2 d3)
    (before d3 d4)
    (before d4 d5)

    (previous d0 d1)
    (previous d0 d2)
    (previous d0 d3)
    (previous d0 d4)
    (previous d0 d5)
    (previous d1 d2)
    (previous d1 d3)
    (previous d1 d4)
    (previous d1 d5)
    (previous d2 d3)
    (previous d2 d4)
    (previous d2 d5)
    (previous d3 d4)
    (previous d3 d5)
    (previous d4 d5)

    (predecessor S0_0 S0_1)
    (predecessor S1_0 S1_1)
    (predecessor S1_1 S1_2)
    (predecessor S2_0 S2_1)
    (predecessor S2_1 S2_2)

    (toWatch S0_1)
```

```
    (toWatch S1_2)
    (toWatch S2_2)
    (toWatch M0)
    (toWatch M1)
    (toWatch M2)

    (parallel M1 S2_0)
    (parallel S2_1 S2_0)

    (= (duration S0_0) 31)
    (= (duration S0_1) 51)
    (= (duration S1_0) 34)
    (= (duration S1_1) 56)
    (= (duration S1_2) 27)
    (= (duration S2_0) 20)
    (= (duration S2_1) 27)
    (= (duration S2_2) 55)
    (= (duration M0) 105)
    (= (duration M1) 74)
    (= (duration M2) 89)

    (= (minutesWatched d0) 0)
    (= (minutesWatched d1) 0)
    (= (minutesWatched d2) 0)
    (= (minutesWatched d3) 0)
    (= (minutesWatched d4) 0)
    (= (minutesWatched d5) 0)
)
```

El estado inicial se asemeja al anterior pero hay ligeras modificaciones para comprobar otro caso. En este estado, los contenidos pendientes (`toWatch`) incluyen los episodios finales de las series y las películas.

Plan encontrado:

```
step    0: WATCH M0 D0
        1: WATCH M2 D0
        2: ADDPARALLELS M1 S2_0
        3: WATCH M1 D1
        4: ADDPREDECESSORS S2_1 S2_2
        5: WATCH S2_0 D1
        6: WATCH S2_1 D2
        7: WATCH S2_2 D3
        8: ADDPREDECESSORS S1_1 S1_2
        9: ADDPREDECESSORS S1_0 S1_1
       10: WATCH S1_0 D2
       11: WATCH S1_1 D3
       12: WATCH S1_2 D4
       13: ADDPREDECESSORS S0_0 S0_1
       14: WATCH S0_0 D2
       15: WATCH S0_1 D3
       16: REACH-GOAL
```

El planificador genera un plan en el que primero se visualizan `M0` y `M2` en el día 0, Al siguiente día, se procesan `M1` y `S2_0`, respetando el paralelismo entre ellos. Los episodios `S2_1` y `S1_0` se completan en el día 2, seguidos de `S2_2` y `S1_1` en el día 3. Finalmente, `S1_2` y `S0_1` se ven en el día 4. Este plan respeta todas las restricciones de precedencia, paralelismo y tiempo, y se completa en 17 pasos.