Documentación de la práctica de Planificación

Laboratorio de Inteligencia Artificial

2º Cuatrimestre - curso 2016/2017

Grau en Informàtica

Departament de Llenguatges i Sistemes Informàtics

Facultat d'Informàtica de Barcelona

UNIVERSITAT POLITÈCNICA DE CATALUNYA

Índice general

Capítulo 1

Organización, evaluación y entrega

Esta es la documentación de la práctica de planificación para los alumnos de Inteligencia Artificial del Grado en Informática. En este documento tenéis:

- Los objetivos de aprendizaje de la práctica correspondientes al temario de la asignatura
- La descripción del problema que debéis resolver
- Lo que tenéis que incluir en el informe que deberéis entregar como resultado de la práctica
- La planificación semanal de la práctica incluyendo los objetivos que debéis ir cubriendo cada semana.

La práctica se debe hacer **preferentemente en grupos de 3 personas**. Intentad no hacerla solos ya que os llevará mucho más trabajo.

La práctica se debe desarrollar con **Fast Forward v2.3**, el planificador que tenéis en la documentación de laboratorio.

Planificad bien el desarrollo de la práctica y no lo dejéis todo para el último día, ya que no seréis capaces de acabarla y hacer un buen trabajo. En este documento tenéis indicaciones sobre el desarrollo de la práctica que os ayudarán a planificar vuestro trabajo.

La valoración de la práctica dependerá de la calidad del modelado del problema, de la cobertura del problema que hagáis, de las extensiones que abordéis y de la calidad de los juegos de prueba.

La entrega de la documentación será el día **5 de junio** en formato electrónico según las instrucciones que aparecerán en el racó.

Capítulo 2

Objetivos de aprendizaje

El objetivo de esta práctica es enfrentarse a un problema sencillo de síntesis que se puede resolver mediante un planificador en el espacio de estados para que construya la solución.

Los objetivos específicos de esta práctica son:

- Implementar mediante un lenguaje de descripción (PDDL) el dominio (predicados y acciones) y varios ejemplos de problemas (objetos, estados inicial y final)
- Aplicar una metodología de desarrollo basada en prototipaje rápido y diseño incremental.
- Saber escoger juegos de prueba suficientemente representativos para demostrar el funcionamiento del sistema y explicar los resultados.
- Tomar contacto con lenguajes de representación de acciones que se puedan usar con planificadores modernos. Se ha de demostrar cierta comprensión y madurez a la hora de utilizar el lenguaje PDDL.
- conectar lo que se ha hecho en la práctica de Sistemas Basados en el Conocimiento con lo que puede hacer el planificador.

Capítulo 3

El problema

Nuestro cliente, después de quedar satisfecho con el sistema de recomendación de menús que le hemos construido en CLIPS, nos pide ahora una funcionalidad especial para poder programar menús semanales en bares y restaurantes.

Un menú esta formado por un primero y un segundo para cada día laborable (de lunes a viernes). Asumiremos que el cocinero es capaz de preparar una serie de primeros y segundos que podremos combinar en los menús. Puede haber incompatibilidades entre primeros y segundos, de manera que no pueden aparecer el mismo día del menú (por ejemplo fideua con all-i-oli de primero y fabada asturiana de segundo). También podemos tener restricciones en el tipo de plato (sopa, crema, ensalada, carne, pescado, ...) entre días consecutivos de manera que este no pueda repetirse. Asumiremos que los platos solo son de un tipo. La restricción la observaremos para primeros y segundos separadamente de manera que no podemos poner dos días seguidos dos primeros platos de pescado (salmón un día y lubina el siguiente), pero puede haber un primero de pescado un día (paella) y un segundo de pescado al día siguiente (salmón). También nos interesa cuidar la dieta de nuestros clientes de manera que disponemos del numero de calorías que tiene cada plato y podemos usarla para obtener combinaciones de platos que aporten las calorías necesarias para una comida saludable. Finalmente, los platos también tienen un coste de elaboración.

Por lo tanto al sistema se le ha de dar conocimiento sobre:

- Platos primeros y segundos que se pueden elaborar
- Tipología a la que pertenece cada plato
- Incompatibilidades entre platos primeros y segundos
- La cantidad de calorías que cada plato aporta
- El precio que cuesta un plato

El resultado es un menú semanal que para cada día nos indica qué primeros y segundos se han de elaborar.

Problema básico y extensiones

- **Nivel básico**: Se obtiene un menú semanal en el que los platos de cada día son compatibles entre si.
- **Extensión 1**: Nivel básico + no se utiliza un mismo plato más de una vez a la semana.
- **Extensión 2**: Extensión 1 + además se cumple la restricción de que no hay días consecutivos con el mismo tipo de plato.

- **Extensión 3**: Extensión 2 + Obligatoriamente ciertos platos específicos han de usarse un día concreto de la semana (por ejemplo paella los jueves)
- **Extensión 4**: Extensión 3 + El planificador controla que, en el menú generado, los platos de un día aporten como mínimo 1000 calorías y como máximo 1500 calorías (en este apartado necesitaréis utilizar Metric-FF).
- **Extensión 5**: Extensión 4 + El planificador minimiza el precio del menú generado (en este apartado necesitaréis utilizar Metric-FF).

Según las extensiones que decidáis abordar la nota de la práctica será diferente:

- Nivel básico: la nota máxima es un 6
- Extensión 1: la nota máxima es un 7
- Extensión 2: la nota máxima es un 8
- Extensión 3: la nota máxima es un 9
- Extensión 4: la nota máxima es un 10
- Extensión 5: la nota máxima es un 11

Nota extra

Los juegos de prueba los podéis hacer a mano, pero se asignará un punto extra a los grupos que hagan un programa (no importa el lenguaje) que pueda generar ficheros con juegos de prueba generados aleatoriamente y se use para hacer los juegos de prueba.

Documentación a entregar

La documentación debe incluir:

- Un documento en el que se describa, de forma razonada
  - La forma en la que se ha modelado el dominio (variables, predicados y acciones)
  - La forma en la que se modelan los problemas a resolver (objetos, estado inicial y final)
  - Una breve explicación de como habeis desarrollado los modelos (de una sola vez, por iteraciones)
  - Un conjunto de problemas de prueba no triviales por cada extensión (mínimo 2), explicando para cada uno que es lo que intentan probar y su resultado. Podéis partir de los juegos de prueba para el nivel básico e ir añadiendo los elementos que cada extensión requiera.
- Código en pddl del dominio que habeis modelado.
- Un fichero que recolecte la traza de la resolucion de los problemas de prueba.

Capítulo 4

Guión de la práctica

Primera semana: Fast Forward/Enunciado/creación del primer prototipo (22 a 28 de mayo)

Esta primera semana la deberéis dedicar a leer el enunciado, a hacer un modelo inicial de dominio y problema y a crear un modelo en PDDL que llegue al nivel básico.

Esta semana se os explicará el funcionamiento del planificador Fast Forward. Es importante que leáis la documentación sobre PDDL y Fast Forward que se os dará, miréis los ejemplos que tenéis e intentéis ejecutarlos.

Tened en cuenta que modelar dominios en PDDL necesita una forma de pensar algo diferente a la que estáis acostumbrados con los lenguajes imperativos y lógicos, por lo que es importante que empecéis cuanto antes a ver cómo funciona.

Si tenéis planeada alguna de las extensiones deberíais de poneros ya con ellas a media semana, ya que la última semana deberéis dedicar algo de tiempo a la documentación y a las pruebas.

En esta práctica es importante planificar vuestro trabajo, no lo dejéis todo para el último momento.

Segunda semana: Prototipo definitivo / Juegos de prueba y documentación (29 de mayo a 4 de junio)

En esta semana deberíais tener ya un planificador que, como mínimo, es capaz de crear planes en el nivel básico. A principios de la semana ya deberíais haber fijado todas las extensiones que queréis intentar hacer y tenerlas algo avanzadas a media semana.

Mirad los ejemplos de problemas modelados en PDDL que tenéis en la web de la asignatura y en otras páginas en Internet para inspiraros.

Deberéis plantearos los casos que queréis probar y mirar que los resultados que esperáis sean los correctos. Haced una lista de casos pensando los diferentes escenarios que es capaz de resolver vuestro sistema.

Pensad que los casos han de ser suficientemente variados tanto en lo que respecta a elementos que intervienen como su complejidad. Tened en cuenta que estos casos os servirán de juegos de prueba, por lo que estáis matando dos pájaros de un tiro. Aprovechad para guardar los resultados y documentarlos.

También deberíais ser capaces de explicar los resultados que obtenéis en función del conocimiento que habéis programado.

Las pruebas deberíais documentarlas adecuadamente explicando cual es el escenario de la prueba y cuales son los resultados que da el sistema.

El resto de la documentación debería explicar todo el proceso de desarrollo y los diferentes prototipos que habéis creado por el camino.

No hace falta que esperéis hasta el último día para entregar. Si acabáis la práctica y la documentación antes podéis entregarla ya durante la semana.

Capítulo 5

Rúbrica de evaluación

Esta es la rúbrica de evaluación de la práctica. La corrección se hará según estos criterios y siguiendo las pautas que se detallan para cada nivel de evaluación.

Deberéis seguir estos criterios a la hora de escribir vuestra documentación y explicar qué habéis hecho en el desarrollo de la práctica y como lo habéis hecho.

Valoraci / Criterio	Mal	Regular	Bien
Dominio	▪ El dominio se representa de manera incompleta o inadecuada (predicados innecesarios)	▪ Se representa completa y adecuadamente las características del dominio ▪ La explicación de la representación del dominio es superficial	▪ Se representa completa y adecuadamente las características del dominio ▪ Se explica detalladamente el significado de cada predicado y se justifica su necesidad
Operadores	▪ El conjunto de operadores es inadecuado o incompleto	▪ El conjunto de operadores es adecuado y completo ▪ La explicación/justificación de los operadores es superficial	▪ El conjunto de operadores es adecuado y completo ▪ Se explica cada operador y se justifica detalladamente su necesidad para la resolución del problema
Juegos de prueba	▪ Juegos de prueba inadecuados para el problema planteado	▪ Juegos de prueba adecuados ▪ No se justifica la elección de los juegos de prueba	▪ Juegos de prueba adecuados ▪ Se justifica la elección de los juegos de prueba ▪ Se explica la solución obtenida
Completado de los niveles	▪ La solución propuesta para los diferentes niveles es inadecuada o incompleta		▪ La solución propuesta para los diferentes niveles es adecuada y completa