



Modelos de Inteligencia Artificial

Profesor/a: Águeda María López Moreno

PRÁCTICA 2.3.- PDDL

26/03/2025

Tabla de contenido

1.- Introducción	4
1.1.- Breve Descripción de la Planificación Automatizada y PDDL	4
1.2.- Importancia de PDDL en la Planificación y la IA	5
1.3.- Objetivos del Trabajo	6
2.- Análisis del Problema y Definiciones Previas	7
2.1.- Entidades del Mundo	7
2.2.- Predicados Disponibles	7
2.3.- Acciones Permitidas	8
2.4.- Escenario Específico	10
2.4.1.- Estado Inicial Detallado (:init)	10
2.4.2.- Estado Objetivo Detallado (:goal)	11
3.- Formalización en PDDL	11
3.1.- Creación del Archivo de Dominio (logistics-domain.pddl)	11
3.1.1.- Definición de Requerimientos y Tipos (:requirements, :types)	12
3.1.2.- Definición de Predicados (:predicates)	14
3.1.3.- Definición de Acciones (:action)	14
3.2.- Creación del Archivo de Problema (transport-problem.pddl)	16
3.2.1.- Vinculación al Dominio (:domain)	16
3.2.2.- Definición de Objetos Específicos (:objects)	17
3.2.3.- Definición del Estado Inicial (:init)	18
4.- Proceso y Herramientas (Cuaderno de Carga)	21
4.1.- PDDL Editor: Definición del Dominio 'logistics'	22
4.2.- PDDL Editor: Definición del Problema 'transporte-practica2-3'	23
4.3.- Resultado de la Ejecución: Plan PDDL	23
4.4.- Resultado final	25
5. Conclusiones	25
5.1. Resumen del Trabajo Realizado	25
5.2. Cumplimiento de los Objetivos	25
5.3. Dificultades Encontradas y Aprendizajes	26
7.- Mapa Mental del Trabajo	27
8.- Anexos	28

8.1.- logistics-domain.pddl	28
8.2.- transport-problem.pddl	28
8.3.- Mapamental.pdf	28
8.4.- Práctica 2.3.- PDDL_Pedro_Manuel_García_Álvarez.docx	29
9.- Bibliografía	29

1.- Introducción

La Planificación Automatizada es un área fundamental de la Inteligencia Artificial (IA) que se ocupa del razonamiento sobre acciones y cambios en el mundo para alcanzar objetivos específicos. Su estudio y aplicación son cruciales en dominios que requieren la toma de decisiones secuenciales, como la robótica, la logística, la manufactura y la gestión de procesos. En este trabajo, se aborda la formalización de problemas de planificación utilizando el **Planning Domain Definition Language (PDDL)**, un lenguaje estándar diseñado específicamente para este propósito, análogo a cómo las formas normales estandarizan expresiones en lógica proposicional.

1.1.- Breve Descripción de la Planificación Automatizada y PDDL

La Planificación Automatizada se enfoca en el desarrollo de algoritmos (planificadores) capaces de generar, de forma autónoma, una secuencia de acciones (un plan) que transforma un estado inicial del mundo en un estado deseado (objetivo), respetando las restricciones y capacidades definidas. Para lograr esto, es esencial contar con un lenguaje formal que permita describir de manera no ambigua los componentes del problema:

- **El Dominio:** Describe la "física" o las reglas generales del entorno. Incluye:
 - **Tipos (Types):** Las categorías de objetos existentes (e.g., camión, paquete, ciudad).
 - **Predicados (Predicates):** Propiedades o relaciones entre objetos que pueden ser verdaderas o falsas en un estado dado (por ejemplo: *en(camión, ciudad)*, *dentro-camión(paquete)*).
 - **Acciones (Actions):** Las operaciones que pueden modificar el estado del mundo, especificando sus parámetros, precondiciones (lo que debe ser cierto para ejecutarla) y efectos (lo que cambia tras su ejecución).

- **El Problema:** Define una instancia específica dentro de ese dominio. Incluye:
 - **Objetos (Objects):** Entidades concretas que participan en el contexto (por ejemplo: *CamionT*, *PaqueteP1*, *CiudadMadrid*).
 - **Estado Inicial (Init):** La descripción completa del mundo al inicio, especificando qué predicados son verdaderos.
 - **Objetivo (Goal):** La condición o conjunto de condiciones que deben ser verdaderas en el estado final.

PDDL surge como respuesta a la necesidad de unificar la forma en que se describen estos dominios y problemas, permitiendo que diferentes sistemas planificadores puedan comprenderlos e intentar resolverlos. Su sintaxis, basada en **LISP**, proporciona una estructura clara para estas definiciones.

1.2.- Importancia de PDDL en la Planificación y la IA

El dominio y uso de PDDL es fundamental por varias razones:

- **Estandarización:** Facilita la comunicación, comparación y evaluación de diferentes algoritmos y sistemas de planificación dentro de la comunidad científica y de desarrollo. Permite la creación de benchmarks y competiciones (como la IPC - International Planning Competition).
- **Separación Conceptual:** Impone una separación clara entre la descripción del modelo del mundo (**dominio**) y la instancia particular a resolver (**problema**), promoviendo la reutilización y modularidad.
- **Precisión y Ausencia de Ambigüedad:** Como lenguaje formal, obliga a definir todos los aspectos relevantes del problema de planificación sin ambigüedades, lo cual es esencial para el razonamiento automático.
- **Puente entre Teoría y Práctica:** Permite aplicar los avances teóricos en algoritmos de planificación a problemas concretos, modelados de forma estándar. Su uso es extendido tanto en

investigación como en aplicaciones industriales incipientes que requieren planificación explícita.

- **Base para Extensiones:** PDDL ha evolucionado para incorporar aspectos más complejos como tiempo, recursos, incertidumbre y preferencias, aunque la práctica actual se centra en su versión básica (STRIPS y tipos).

1.3.- Objetivos del Trabajo

El objetivo principal de este trabajo es aplicar los conceptos de la planificación automatizada mediante la utilización del lenguaje PDDL para modelar un escenario logístico específico. Los objetivos específicos son los siguientes:

1. **Comprender la estructura y sintaxis de PDDL:** Familiarizarse con los componentes clave de un archivo de dominio (:requirements, :types, :predicates, :action) y un archivo de problema (:objects, :init, :goal).
2. **Analizar y formalizar un problema de planificación:** Traducir la descripción en lenguaje natural de un problema logístico (entidades, restricciones, acciones, estado inicial y objetivo) a las estructuras formales de PDDL.
3. **Desarrollar los archivos PDDL:** Crear el archivo de dominio (logistics-domain.pddl) que define las reglas generales del transporte de paquetes y el archivo de problema (transport-problem.pddl) que especifica la instancia concreta a resolver (paquetes P1 y P2, ciudades Barcelona, Madrid, Sevilla, y camión T).
4. **Documentar el proceso de modelado:** Registrar detalladamente los pasos seguidos, las decisiones tomadas y el código PDDL resultante, sirviendo este documento como evidencia del proceso y resultado (cuaderno de carga).

2.- Análisis del Problema y Definiciones Previas

Antes de proceder a la formalización del problema de planificación logística en **PDDL**, es fundamental realizar un análisis detallado de los componentes del mundo, las propiedades que describen su estado, las operaciones que pueden alterarlo y el escenario específico que se busca resolver. Esta sección desglosa estos elementos constitutivos.

2.1.- Entidades del Mundo

El dominio del problema está poblado por distintas categorías de entidades u objetos, cuya interacción define el sistema logístico. En el formalismo **PDDL**, estas categorías se representan mediante **tipos (types)**. Para este problema, se identifican los siguientes tipos fundamentales:

- **paquete**: Representa los objetos que deben ser transportados. Son entidades móviles que pueden encontrarse en una ciudad o dentro del camión.
- **ciudad**: Representa las localizaciones geográficas discretas entre las cuales ocurre el transporte. Son puntos fijos en el mapa del problema.
- **camión**: Representa el agente activo del sistema, responsable de realizar el transporte. El camión tiene una ubicación y un estado de carga (vacío o no).

2.2.- Predicados Disponibles

Los predicados definen las propiedades de los objetos y las relaciones entre ellos que pueden ser verdaderas (**true**) o falsas (**false**) en un estado determinado del mundo. Describen la configuración del

sistema en cualquier instante. Para este dominio, se definen los siguientes predicados (siguiendo la sintaxis PDDL):

- **autovia(?c1 - ciudad ?c2 - ciudad)**: Predicado estático (su valor no cambia por las acciones) que indica si existe una conexión directa (autovía) entre la ciudad ?c1 y la ciudad ?c2. Es una relación binaria entre ciudades.
- **en-camion(?t - camion ?c - ciudad)**: Predicado dinámico que es verdadero si el camión ?t se encuentra actualmente en la ciudad ?c. Define la ubicación del agente transportista.
- **en-paquete(?p - paquete ?c - ciudad)**: Predicado dinámico que es verdadero si el paquete ?p se encuentra actualmente en la ciudad ?c. Este predicado es falso si el paquete está dentro del camión. Define la ubicación de un paquete cuando está en una localización fija.
- **dentro-camion(?p - paquete ?t - camion)**: Predicado dinámico que es verdadero si el paquete ?p está cargado dentro del camión ?t. Es mutuamente excluyente con en-paquete(?p, ?c) para el mismo paquete ?p.
- **descargado(?t - camion)**: Predicado dinámico que es verdadero si el camión ?t no contiene ningún paquete en su interior (está vacío).

(Nota: Las descripciones originales mencionaban PAQUETE(x) y CIUDAD(x). En PDDL, estas características se manejan mediante la asignación de **tipos** a los objetos, no como predicados de estado dinámicos. El predicado EN(x,c) original se desglosa en en-camion y en-paquete para mayor precisión según el tipo de objeto x).

2.3.- Acciones Permitidas

Las acciones representan las operaciones que el agente (el camión) puede realizar para cambiar el estado del mundo. Cada acción se define por sus parámetros, las precondiciones que deben cumplirse para poder ejecutarla y los efectos que produce sobre el estado (qué predicados se vuelven verdaderos y/o falsos).

- **CARGA(?p - paquete, ?c - ciudad, ?t - camion)**:

- **Propósito:** El camión ?t carga el paquete ?p en la ciudad ?c.
- **Precondiciones:**
 - El camión ?t debe estar en la ciudad ?c (en-camion ?t ?c).
 - El paquete ?p debe estar en la ciudad ?c (en-paquete ?p ?c).
 - El camión ?t debe estar descargado (descargado ?t).
- **Efectos:**
 - El paquete ?p pasa a estar dentro del camión ?t (dentro-camion ?p ?t).
 - El camión ?t ya no está descargado (not (descargado ?t)).
 - El paquete ?p ya no está en la ciudad ?c (not (en-paquete ?p ?c)).
- **DESCARGA(?p - paquete, ?c - ciudad, ?t - camion):**
 - **Propósito:** El camión ?t descarga el paquete ?p en la ciudad ?c.
 - **Precondiciones:**
 - El camión ?t debe estar en la ciudad ?c (en-camion ?t ?c).
 - El paquete ?p debe estar dentro del camión ?t (dentro-camion ?p ?t).
 - **Efectos:**
 - El paquete ?p pasa a estar en la ciudad ?c (en-paquete ?p ?c).
 - El camión ?t pasa a estar descargado (descargado ?t).
 - El paquete ?p ya no está dentro del camión ?t (not (dentro-camion ?p ?t)).
- **IR(?c1 - ciudad, ?c2 - ciudad, ?t - camion):**
 - **Propósito:** El camión ?t se desplaza desde la ciudad ?c1 a la ciudad ?c2.
 - **Precondiciones:**
 - El camión ?t debe estar en la ciudad origen ?c1 (en-camion ?t ?c1).
 - Debe existir una autovía entre ?c1 y ?c2 (autovia ?c1 ?c2).
 - **Efectos:**

- El camión ?t pasa a estar en la ciudad destino ?c2 (en-camion ?t ?c2).
- El camión ?t ya no está en la ciudad origen ?c1 (not (en-camion ?t ?c1)).

2.4.- Escenario Específico

Para este trabajo, se define una instancia concreta del problema de planificación logística, detallando los objetos particulares, el estado inicial del mundo y el estado objetivo deseado.

2.4.1.- Estado Inicial Detallado (:init)

La configuración inicial del sistema es la siguiente:

- **Objetos Concretos:**
 - Paquetes: P1, P2 (de tipo paquete).
 - Ciudades: Barcelona, Madrid, Sevilla (de tipo ciudad).
 - Camión: T (de tipo camion).
- **Predicados Verdaderos al Inicio:**
 - Ubicación Paquetes:
 - (en-paquete P1 Barcelona)
 - (en-paquete P2 Madrid)
 - Ubicación Camión:
 - (en-camion T Sevilla)
 - Estado Camión:
 - (descargado T) (Se asume que el camión comienza vacío).
 - Conexiones de Autovía:
 - (autovia Barcelona Madrid)
 - (autovia Madrid Barcelona) (*Necesario si la acción IR no asume simetría*)
 - (autovia Madrid Sevilla)
 - (autovia Sevilla Madrid) (*Necesario si la acción IR no asume simetría*)

2.4.2.- Estado Objetivo Detallado (:goal)

El objetivo final que se desea alcanzar es una configuración del mundo donde se cumplan simultáneamente las siguientes condiciones:

- **(en-paquete P1 Sevilla)**: El paquete **P1** debe encontrarse en la ciudad de Sevilla.
- **(en-paquete P2 Barcelona)**: El paquete **P2** debe encontrarse en la ciudad de Barcelona.
- **(descargado T)**: El camión **T** debe estar descargado (sin paquetes en su interior).

La tarea del planificador será encontrar una secuencia válida de acciones (**CARGA**, **DESCARGA**, **IR**) que transforme el estado inicial descrito en 2.4.1 en un estado donde las condiciones del objetivo 2.4.2 sean todas verdaderas.

3.- Formalización en PDDL

Una vez analizados los componentes del problema de planificación, el siguiente paso es traducirlos al lenguaje formal **PDDL**. Este proceso implica la creación de dos archivos principales: el archivo de dominio, que establece las reglas generales del mundo logístico, y el archivo de problema, que define la instancia específica a resolver.

3.1.- Creación del Archivo de Dominio (logistics-domain.pddl)

El archivo de dominio encapsula la estructura general del entorno de planificación. Define los tipos de objetos que existen, las propiedades y relaciones (**predicados**) que describen el estado, y las acciones que pueden modificar dicho estado.

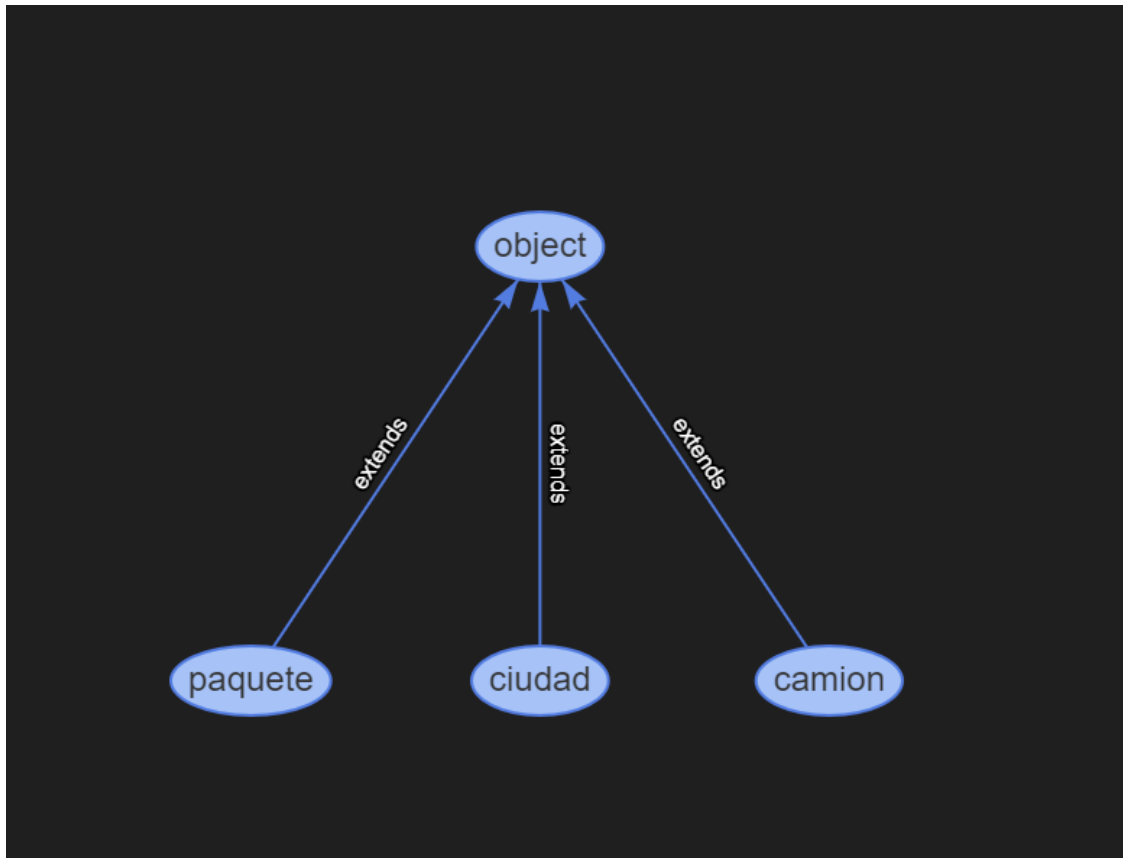
```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;  
; Dominio PDDL para el Problema de Planificación Logística (Práctica 2-3)  
; Descripción: Define los tipos, predicados y acciones para un sistema  
;             de transporte de paquetes mediante un camión entre ciudades.  
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;  
  
(define (domain logistics)
```

Este código define un dominio **PDDL** llamado "**logistics**" que establece la estructura y las reglas para un problema de planificación logística, incluyendo tipos, predicados y acciones relacionadas con el transporte de paquetes entre ciudades mediante un camión.

3.1.1.- Definición de Requerimientos y Tipos (:requirements, :types)

```
; Requerimientos: Habilitamos la sintaxis básica de STRIPS y la definición de tipos.  
(:requirements :strips :typing)
```

Este código habilita los requerimientos básicos de **STRIPS** y la capacidad de definir tipos en un dominio **PDDL**, permitiendo el uso de la sintaxis fundamental de planificación y la especificación de tipos para objetos y parámetros.



La imagen representa un diagrama jerárquico donde "paquete", "ciudad" y "camión" son subclases que extienden la clase "object".

```
; Tipos: Definición de las categorías de objetos existentes en el mundo.
(:types
  paquete ; Objetos a transportar
  ciudad ; Localizaciones fijas
  camion ; Agente de transporte
)
```

Este código define tres tipos de objetos en el dominio PDDL: **paquete**, **ciudad** y **camión**, estableciendo las categorías fundamentales para el problema de logística.

3.1.2.- Definición de Predicados (:predicates)

```

; Predicados: Propiedades y relaciones que describen el estado del mundo.
(:predicates
  (autovia ?c1 - ciudad ?c2 - ciudad)      ; Estático: Indica si hay conexión directa entre c1 y c2.
  (en-camion ?t - camion ?c - ciudad)       ; Dinámico: El camión ?t está en la ciudad ?c.
  (en-paquete ?p - paquete ?c - ciudad)     ; Dinámico: El paquete ?p está en la ciudad ?c (no en el camión).
  (dentro-camion ?p - paquete ?t - camion)  ; Dinámico: El paquete ?p está dentro del camión ?t.
  (descargado ?t - camion)                  ; Dinámico: El camión ?t está vacío.
)

```

Este código define los predicados que describen las propiedades y relaciones del mundo en el problema de logística, incluyendo conexiones entre **ciudades**, ubicaciones de **camiones** y **paquetes**, y el estado de carga de los **camiones**.

3.1.3.- Definición de Acciones (:action)

3.1.3.1.- ACCIÓN CARGA

```

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; Acciones: Operaciones que modifican el estado del mundo.
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

; Acción: Cargar un paquete en el camión.
(:action CARGA
  :parameters (?p - paquete ?c - ciudad ?t - camion)
  :precondition (and
    (en-camion ?t ?c)      ; El camión debe estar en la ciudad.
    (en-paquete ?p ?c)     ; El paquete debe estar en la misma ciudad.
    (descargado ?t)        ; El camión debe estar vacío para cargar.
  )
  :effect (and
    (dentro-camion ?p ?t)  ; El paquete ahora está dentro del camión.
    (not (descargado ?t))  ; El camión ya no está vacío.
    (not (en-paquete ?p ?c)) ; El paquete ya no está en la ciudad.
  )
)

```

Este código define la acción "**CARGA**" en el dominio **PDDL**, que permite cargar un paquete en un camión en una ciudad específica, estableciendo las precondiciones necesarias y los efectos resultantes de la acción.

3.1.3.2. ACCIÓN DESCARGA

```
; Acción: Descargar un paquete del camión.
(:action DESCARGA
 :parameters (?p - paquete ?c - ciudad ?t - camion)
 :precondition (and
   (en-camion ?t ?c)      ; El camión debe estar en la ciudad de descarga.
   (dentro-camion ?p ?t) ; El paquete debe estar dentro del camión.
 )
 :effect (and
   (en-paquete ?p ?c)      ; El paquete ahora está en la ciudad.
   (descargado ?t)         ; El camión vuelve a estar descargado.
   (not (dentro-camion ?p ?t)); El paquete ya no está dentro del camión.
 )
)
```

Este código define la acción "**DESCARGA**" en el dominio **PDDL**, que permite descargar un **paquete** de un **camión** en una **ciudad** específica, estableciendo las precondiciones necesarias y los efectos resultantes de la acción.

3.1.3.3. ACCIÓN IR

```
; Acción: Mover el camión entre ciudades conectadas por autovía.
(:action IR
 :parameters (?c1 - ciudad ?c2 - ciudad ?t - camion) ; Origen c1, Destino c2, Camión t
 :precondition (and
   (en-camion ?t ?c1)      ; El camión debe estar en la ciudad origen c1.
   (autovia ?c1 ?c2)       ; Debe existir una autovía directa de c1 a c2.
 )
 :effect (and
   (en-camion ?t ?c2)      ; El camión ahora está en la ciudad destino c2.
   (not (en-camion ?t ?c1)) ; El camión ya no está en la ciudad origen c1.
 )
)
```

Este código define la acción "**IR**" en el dominio **PDDL**, que permite mover un **camión** entre dos **ciudades** conectadas por una **autovía**, estableciendo las precondiciones necesarias y los efectos resultantes del movimiento.

3.2.- Creación del Archivo de Problema (transport-problem.pddl)

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;  
; Problema PDDL para la Práctica 2-3: Transporte de Paquetes  
; Descripción: Define la instancia específica del problema logístico,  
;             incluyendo objetos, estado inicial y objetivo.  
; Objetivo: Mover P1 de Barcelona a Sevilla y P2 de Madrid a Barcelona.  
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;  
  
(define (problem transporte-practica2-3)
```

Este código define un problema específico de transporte de paquetes en **PDDL**, estableciendo los **objetos**, el estado inicial y los objetivos para mover dos **paquetes** entre diferentes **ciudades**.

3.2.1.- Vinculación al Dominio (:domain)

```
; Dominio: Especifica el dominio PDDL que define las reglas de este problema.  
(:domain logistics)
```

Este código especifica que el problema utiliza el dominio **"logistics"** previamente definido, vinculando así el problema con las reglas y acciones establecidas en ese dominio **PDDL**.

3.2.2.- Definición de Objetos Específicos (:objects)



La imagen representa una jerarquía de objetos donde '**paquete**', '**ciudad**', y '**camión**' extienden '**object**', y cada uno está instanciado con ejemplos como '**P1**', '**P2**', '**Barcelona**', '**Madrid**', '**Sevilla**', y '**T**' respectivamente.

```
; Objetos: Declaración de las entidades concretas de esta instancia.
(:objects
  P1 P2      - paquete      ; Los dos paquetes a transportar
  Barcelona Madrid Sevilla - ciudad ; Las tres ciudades involucradas
  T          - camion      ; El único camión disponible
)
```

Este código declara los objetos específicos del problema: dos **paquetes** (**P1** y **P2**), tres **ciudades** (**Barcelona**, **Madrid** y **Sevilla**) y un **camión** (**T**), asignándoles sus respectivos tipos definidos en el dominio.

3.2.3.- Definición del Estado Inicial (:init)

PLAN VISUALIZATION

value	
camion	descargado
T	true

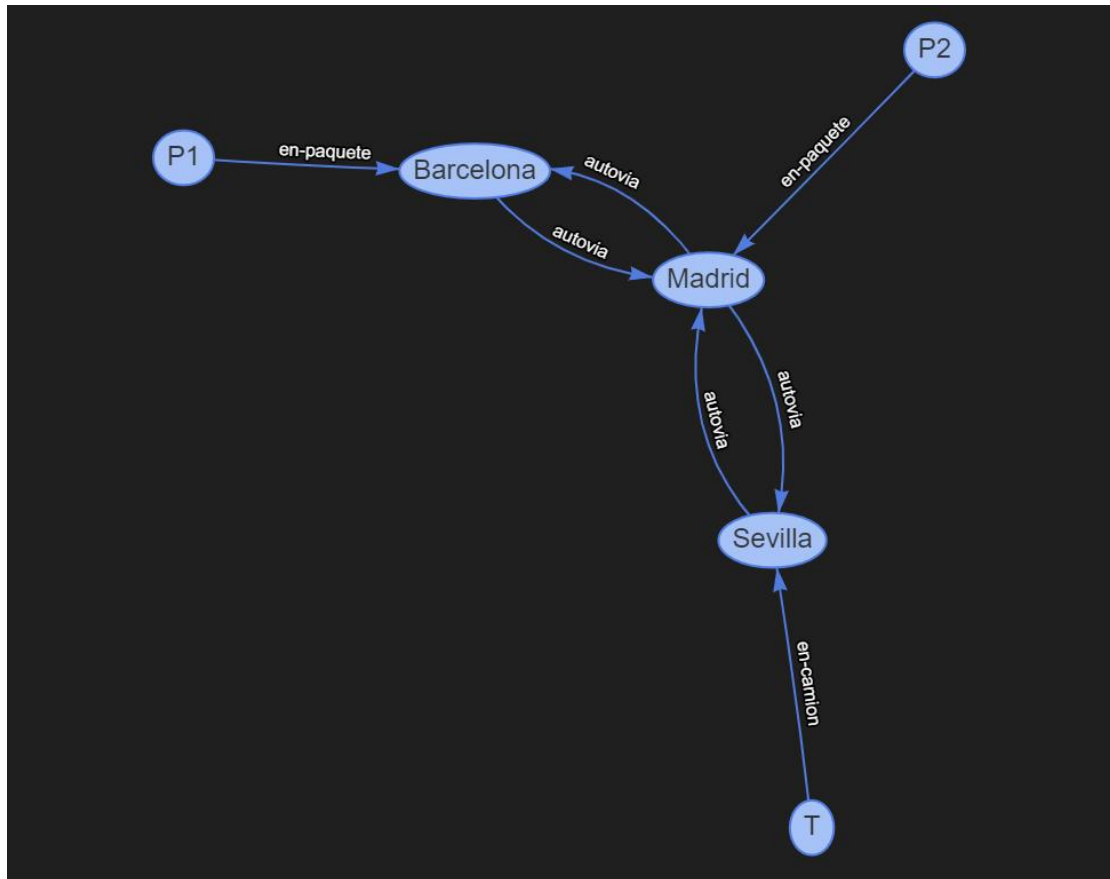
ciudad	Barcelona	Madrid	Sevilla
Barcelona		autovia	
Madrid	autovia		autovia
Sevilla		autovia	

camion \ ciudad	Barcelona	Madrid	Sevilla
T			en-camion

paquete \ ciudad	Barcelona	Madrid	Sevilla
P1	en-paquete		
P2		en-paquete	

paquete \ camion	T
P1	
P2	

La imagen representa una visualización de un plan logístico que involucra un **camión**, **paquetes** y **ciudades**, mostrando las relaciones entre ellos, como el estado del camión (**descargado**), las rutas entre **ciudades** (**autovía**), la ubicación de los **paquetes** (**en-paquete**) y el estado de cada **camión** (**en-camion**).



La imagen muestra un diagrama de red que representa la logística de **paquetes** (P1, P2) entre **ciudades** (Barcelona, Madrid, Sevilla) conectadas por **autopistas** y el estado de un **camión** (T) en Sevilla.

```
; Estado Inicial: Describe la configuración inicial del mundo.
(:init
  ; Ubicación inicial de los paquetes
  (en-paquete P1 Barcelona)
  (en-paquete P2 Madrid)

  ; Ubicación inicial del camión
  (en-camion T Sevilla)

  ; Estado inicial del camión (vacío)
  (descargado T)

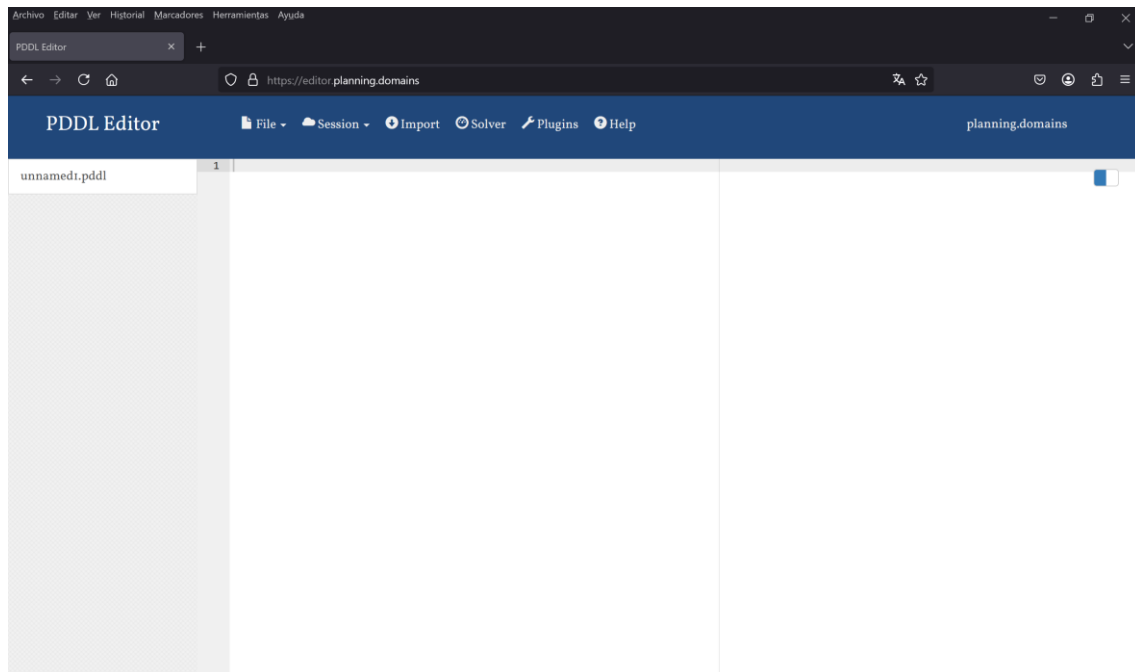
  ; Conexiones de autovía (definidas bidireccionalmente)
  (autovia Barcelona Madrid)
  (autovia Madrid Barcelona)
  (autovia Madrid Sevilla)
  (autovia Sevilla Madrid)
)
```

Este código define el estado inicial del problema, especificando las **ubicaciones** de los **paquetes** y el **camión**, el estado de carga del **camión**, y las conexiones de **autovía** entre las **ciudades**.

```
; Objetivo: Describe el estado final deseado.
(:goal (and
  (en-paquete P1 Sevilla) ; P1 debe terminar en Sevilla.
  (en-paquete P2 Barcelona) ; P2 debe terminar en Barcelona.
  (descargado T) ; El camión T debe terminar descargado.
)
)
```

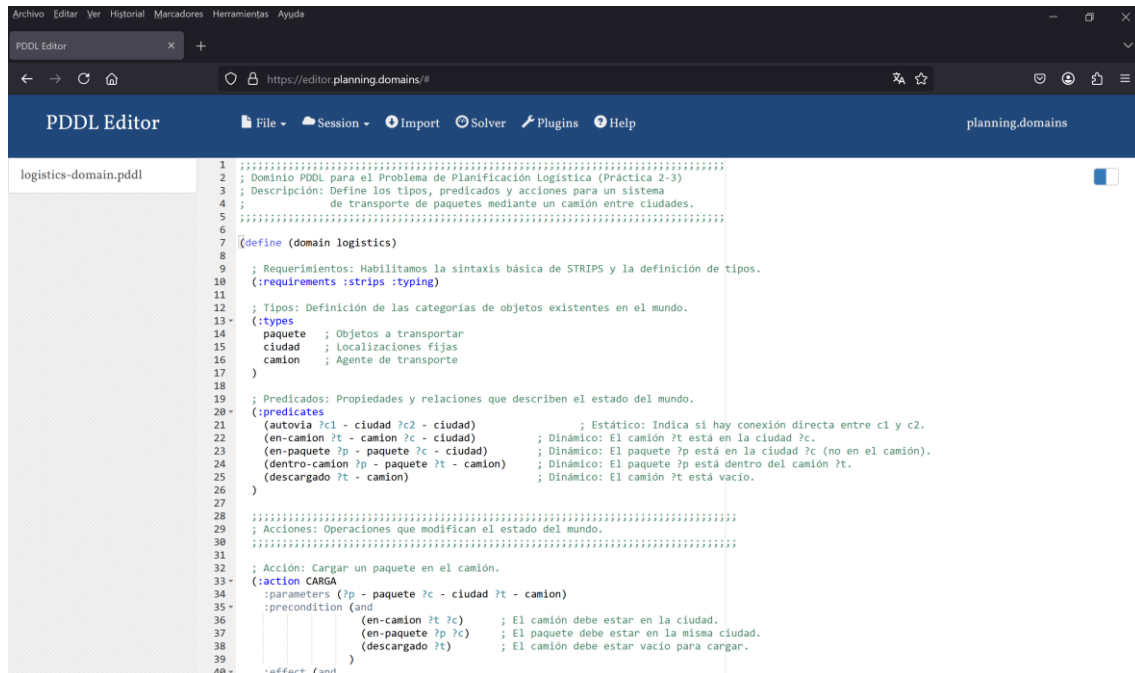
Este código define el objetivo del problema, especificando que el **paquete P1** debe estar en **Sevilla**, el **paquete P2** en **Barcelona**, y el **camión T** debe terminar descargado.

4.- Proceso y Herramientas (Cuaderno de Carga)



Esta pantalla muestra el **Editor PDDL online**, una herramienta de desarrollo utilizada en campos como la **Inteligencia Artificial, la robótica y la logística**. Sirve para **definir formalmente problemas de planificación**: se describe el estado del mundo, las acciones posibles y el objetivo deseado, utilizando el lenguaje estándar **PDDL**. Estos modelos son luego procesados por '**solvers**' para encontrar automáticamente la secuencia óptima de acciones para alcanzar dicho objetivo, permitiendo **automatizar la toma de decisiones complejas** en sistemas autónomos o procesos operativos.

4.1.- PDDL Editor: Definición del Dominio 'logistics'



```
1 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
2 ; Dominio PDDL para el Problema de Planificación Logística (Práctica 2-3)
3 ; Descripción: Define los tipos, predicados y acciones para un sistema
4 ; de transporte de paquetes mediante un camión entre ciudades.
5 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
6
7 (define (domain logistics)
8
9   ; Requerimientos: Habilitamos la sintaxis básica de STRIPS y la definición de tipos.
10  (:requirements :strips :typing)
11
12  ; Tipos: Definición de las categorías de objetos existentes en el mundo.
13  (:types
14    paquete ; Objetos a transportar
15    ciudad ; Localizaciones fijas
16    camion ; Agente de transporte
17  )
18
19  ; Predicados: Propiedades y relaciones que describen el estado del mundo.
20  (:predicates
21    (autovia ?c1 - ciudad ?c2 - ciudad) ; Estático: Indica si hay conexión directa entre c1 y c2.
22    (en-camion ?t - camion ?c - ciudad) ; Dinámico: El camión ?t está en la ciudad ?c.
23    (en-paquete ?p - paquete ?c - ciudad) ; Dinámico: El paquete ?p está en la ciudad ?c (no en el camión).
24    (dentro-camion ?p - paquete ?t - camion) ; Dinámico: El paquete ?p está dentro del camión ?t.
25    (descargado ?t - camion) ; Dinámico: El camión ?t está vacío.
26  )
27
28  ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
29  ; Acciones: Operaciones que modifican el estado del mundo.
30  ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
31
32  ; Acción: Cargar un paquete en el camión.
33  (:action CARGA
34    :parameters (?p - paquete ?c - ciudad ?t - camion)
35    :precondition (and
36      (en-camion ?t ?c) ; El camión debe estar en la ciudad.
37      (en-paquete ?p ?c) ; El paquete debe estar en la misma ciudad.
38      (descargado ?t) ; El camión debe estar vacío para cargar.
39    )
40    :effect (and
```

Esta pantalla muestra el **Editor PDDL online** (<https://editor.planning.domains/>) donde se está visualizando y editando el **dominio logistics**, que define los tipos de objetos (**paquete**, **ciudad**, **camión**), sus estados (**predicados**) y las acciones (**CARGA**, **DESCARGA**, **IR**) para resolver problemas de planificación de transporte.

4.2.- PDDL Editor: Definición del Problema 'transporte-practica2-3'

```

1 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
2 ; Problema PDDL para la Práctica 2-3: Transporte de Paquetes
3 ; Descripción: Define la instancia específica del problema logístico,
4 ; incluyendo objetos, estado inicial y objetivo.
5 ; Objetivo: Mover P1 de Barcelona a Sevilla y P2 de Madrid a Barcelona.
6 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
7
8 (define (problem transporte-practica2-3)
9
10 ; Dominio: Especifica el dominio PDDL que define las reglas de este problema.
11 (:domain logistics)
12
13 ; Objetos: Declaración de las entidades concretas de esta instancia.
14 (:objects
15   P1 P2          - paquete           ; Los dos paquetes a transportar
16   Barcelona Madrid Sevilla - ciudad ; Las tres ciudades involucradas
17   T              - camion           ; El único camión disponible
18 )
19
20 ; Estado Inicial: Describe la configuración inicial del mundo.
21 (:init
22   ; Ubicación inicial de los paquetes
23   (en-paquete P1 Barcelona)
24   (en-paquete P2 Madrid)
25
26   ; Ubicación inicial del camión
27   (en-camion T Sevilla)
28
29   ; Estado inicial del camión (vacío)
30   (descargado T)
31
32   ; Conexiones de autovía (definidas bidireccionalmente)
33   (autovia Barcelona Madrid)
34   (autovia Madrid Barcelona)
35   (autovia Madrid Sevilla)
36   (autovia Sevilla Madrid)
37 )
38
39 ; Objetivo: Describe el estado final deseado.
40 (:goal (and

```

Esta pantalla muestra el **Editor PDDL online** (<https://editor.planning.domains/>) presentando el código PDDL que define el problema específico **transporte-practica2-3**, detallando los objetos (**paquetes**, **ciudades**, **camión**), su estado inicial y el objetivo final de mover los **paquetes** a sus destinos designados.

4.3.- Resultado de la Ejecución: Plan PDDL

Compute plan

Domain: logistics-domain.pddl

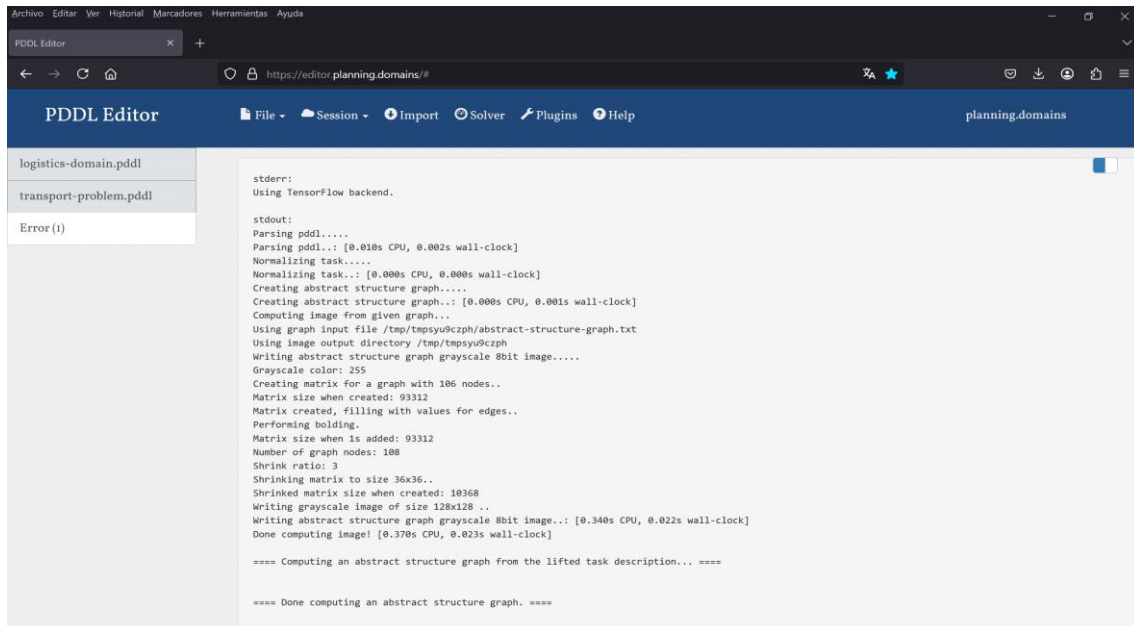
Problem: transport-problem.pddl

Solver: Delfi: online planner selection

Description: This planner uses an offline learned algorithm selector based on a abstract structure graph of the PDDL description of the planning task.

Plan **Cancel**

En esta pantalla, se procede a ejecutar el código para que pueda solucionar el problema pulsando en el menú superior en la opción “**Solver**”, luego pulsamos el botón “**Plan**” para proceder a la ejecución del problema.



```

stderr:
Using TensorFlow backend.

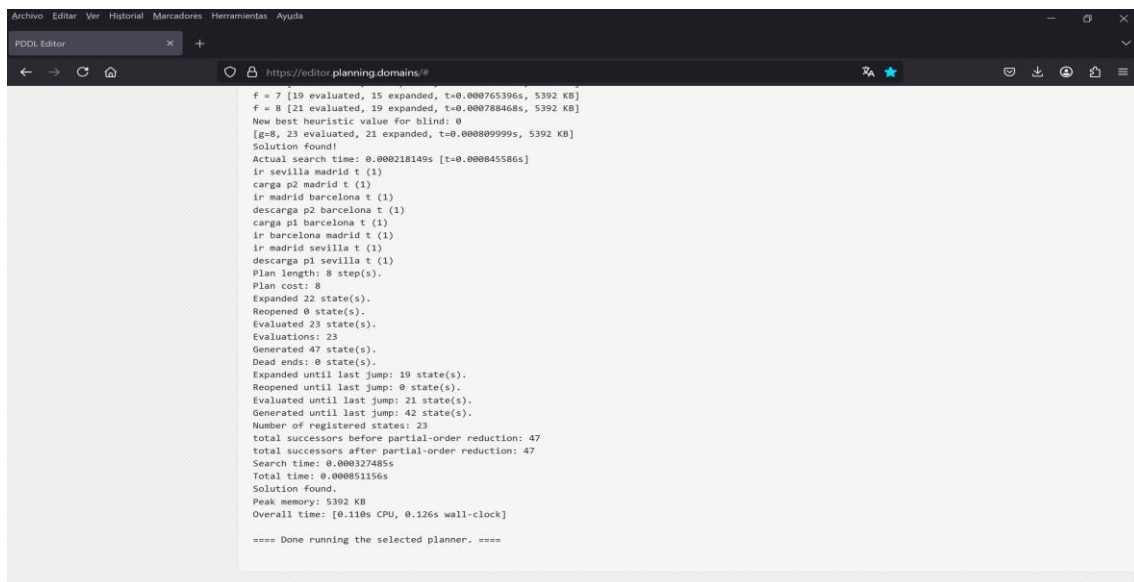
stdout:
Parsing pddl.....
Parsing pddl... [0.018s CPU, 0.002s wall-clock]
Normalizing task.....
Normalizing task... [0.000s CPU, 0.000s wall-clock]
Creating abstract structure graph.....
Creating abstract structure graph... [0.000s CPU, 0.001s wall-clock]
Computing image from given graph...
Using graph input file /tmp/tmpsydczph/abstract-structure-graph.txt
Using image output directory /tmp/tmpsydczph
Writing abstract structure graph grayscale 8bit image.....
Grayscale color: 255
Creating matrix for a graph with 106 nodes..
Matrix size when created: 93312
Matrix created, filling with values for edges..
Performing holding.
Matrix size when is added: 93312
Number of graph nodes: 108
Shrink ratio: 3
Shrinking matrix to size 36x36..
Shrunk matrix size when created: 10368
Writing grayscale image of size 128x128 ..
Writing abstract structure graph grayscale 8bit image... [0.340s CPU, 0.022s wall-clock]
Done computing image! [0.370s CPU, 0.023s wall-clock]

==== Computing an abstract structure graph from the lifted task description... ====

==== Done computing an abstract structure graph. ====

```

La pantalla exhibe un editor en línea para planificar tareas, acompañado de un registro detallado del proceso que incluye la creación de un grafo abstracto y la selección de un planificador.



```

f = 7 [19 evaluated, 15 expanded, t=0.000765396s, 5392 KB]
f = 8 [21 evaluated, 19 expanded, t=0.000788468s, 5392 KB]
New best heuristic value for blind: 0
[g=8, 23 evaluated, 21 expanded, t=0.000809999s, 5392 KB]
Solution found!
Actual search time: 0.000218149s [t=0.000845586s]
ir sevilla madrid t (1)
carga p2 madrid t (1)
ir madrid barcelona t (1)
descarga p2 barcelona t (1)
carga p1 barcelona t (1)
ir barcelona madrid t (1)
ir madrid sevilla t (1)
descarga p1 sevilla t (1)
Plan length: 8 step(s).
Plan cost: 8
Expanded 22 state(s).
Reopened 0 state(s).
Evaluated 23 state(s).
Evaluations: 23
Generated 47 state(s).
Dead ends: 0 state(s).
Expanded until last jump: 19 state(s).
Reopened until last jump: 0 state(s).
Evaluated until last jump: 21 state(s).
Generated until last jump: 42 state(s).
Number of registered states: 23
total successors before partial-order reduction: 47
total successors after partial-order reduction: 47
Search time: 0.000327485s
Total time: 0.000851156s
Solution found.
Peak memory: 5392 KB
Overall time: [0.110s CPU, 0.126s wall-clock]

==== Done running the selected planner. ====

```

Esta pantalla muestra el resultado de la ejecución de un planificador, indicando que encontró una solución con un plan de 8 pasos, junto con diversas métricas sobre la búsqueda realizada.

4.4.- Resultado final



El resultado muestra un diagrama de flujo centralizado que desglosa el plan de acción en pasos numerados y codificados por color, representando la secuencia de movimientos y **cargas/descargas** del transporte y los **paquetes** entre **ciudades**.

5. Conclusiones

5.1. Resumen del Trabajo Realizado

En este trabajo, se analizó el resultado de un proceso de planificación automatizada utilizando una combinación de técnicas de aprendizaje automático (para la selección del planificador) y planificación clásica (con **Fast Downward**). Inicialmente, se revisó el log de ejecución, identificando las etapas clave: **parsing** del problema **PDDL**, creación de un grafo de estructura abstracta (**ASG**), selección de un planificador mediante un modelo de aprendizaje automático, y finalmente, la ejecución del planificador **Fast Downward**. Este último generó un plan de 8 pasos para un problema de transporte, detallando las acciones de movimiento (**ir**) y manipulación de **paquetes** (**carga**, **descarga**) entre las **ciudades** de **Sevilla**, **Madrid** y **Barcelona**, utilizando un transporte específico **t (1)** y dos paquetes **p1** y **p2**. Posteriormente, se exploraron diversas formas de representar visualmente este plan, desde una lista ordenada en **Markdown** hasta un **diagrama de flujo**, utilizando herramientas como la sintaxis de **Mermaid** y potencialmente extensiones de **Visual Studio Code** como **Draw.io** para una representación gráfica más elaborada.

5.2. Cumplimiento de los Objetivos

El objetivo principal de este ejercicio era comprender el proceso de planificación automatizada a través del análisis de un log de ejecución y la visualización del plan resultante. Se logró analizar el log, identificando las diferentes fases del proceso y el planificador seleccionado y ejecutado. Además, se cumplió el objetivo de representar el plan de manera textual (en **Markdown**) y visual (a través de descripciones para generar un grafo y el uso de la sintaxis **Mermaid**), lo que permitió una comprensión más intuitiva de la secuencia de acciones necesarias para resolver el problema de planificación. La exploración de posibles herramientas en **Visual Studio Code** para la creación de grafos también contribuyó a una comprensión más amplia de las opciones disponibles para la visualización de planes y procesos.

5.3. Dificultades Encontradas y Aprendizajes

Una dificultad inicial fue interpretar el log de ejecución sin un conocimiento previo detallado del sistema **DELFI y Fast Downward**. Sin embargo, a través del análisis secuencial de las líneas de salida, se logró comprender el flujo general del proceso, desde la lectura del problema hasta la obtención de la solución.

El principal aprendizaje fue la apreciación de la complejidad inherente a los sistemas de planificación híbridos que combinan aprendizaje automático y planificación clásica. Se observó cómo el **ASG** sirve como una representación abstracta del problema para el modelo de aprendizaje, que a su vez influye en la elección del planificador. También se comprendió la importancia de la representación visual para facilitar la comprensión de un plan secuencial de acciones. La exploración de diferentes formatos (**Markdown**, sintaxis **Mermaid**) y herramientas (extensiones de **VS Code**) demostró la variedad de opciones disponibles para comunicar y analizar los resultados de la planificación automatizada.

Aunque no se encontraron errores en el proceso de planificación en sí (según el log), la mención de un "Error (1)" en la interfaz del editor online generó una breve incertidumbre, resaltando la

importancia de verificar los resultados y el contexto en diferentes capas del sistema.

7.- Mapa Mental del Trabajo



El mapa mental del trabajo proporciona una representación visual estructurada que ilustra los elementos clave y las relaciones dentro del proceso de análisis y planificación realizado, ofreciendo una visión general de su estructura.

8.- Anexos

Los anexos adjuntos complementan la documentación principal, aportando los documentos y archivos fuente esenciales para una comprensión completa del trabajo realizado en el campo de la planificación automatizada.

8.1.- logistics-domain.pddl

Este fichero presenta la definición formal del dominio de planificación "**logistics**" utilizando el Lenguaje de Definición de Dominios de Planificación (**PDDL**). Su contenido articula el vocabulario esencial del entorno del problema, detallando los tipos de entidades, las propiedades y relaciones relevantes a través de predicados, y las acciones permisibles mediante la especificación de operadores con sus respectivas precondiciones y efectos.

8.2.- transport-problem.pddl

Se incluye en este fichero define una instancia específica del problema de transporte dentro del dominio "**logistics**" previamente establecido en **logistics-domain.pddl**. Este documento particulariza el escenario a resolver por el planificador, declarando los objetos concretos que intervienen, describiendo la configuración inicial del mundo mediante el conjunto de predicados verdaderos, y formalizando el estado objetivo que se pretende alcanzar a través de la planificación.

8.3.- Mapamental.pdf

Este fichero contiene un mapa mental elaborado como herramienta de apoyo durante las fases de conceptualización,

organización de ideas o visualización del plan resultante. Este diagrama gráfico facilita la comprensión de las relaciones entre los elementos clave del problema de planificación o la estructura del plan generado.

8.4.- Práctica 2.3.- PDDL_Pedro_Manuel_García_Álvarez.docx

El presente fichero corresponde al documento principal de la trabajo o ejercicio académico titulado "PDDL". Este archivo en formato DOCX alberga la documentación exhaustiva del trabajo realizado, incluyendo la introducción, los objetivos, la descripción detallada del problema, las especificaciones PDDL completas de los archivos de dominio y problema, la exposición del proceso de planificación seguido, el análisis de los resultados obtenidos, las conclusiones derivadas del estudio y, en su caso, posibles líneas de trabajo futuro.

9.- Bibliografía

La siguiente bibliografía compila los recursos académicos, técnicos y de software referenciados o utilizados en la elaboración de este trabajo sobre planificación automatizada. Se incluyen enlaces para facilitar el acceso a la documentación y fuentes originales.

- **Ghallab, M., Nau, D., & Traverso, P. (2016). *Automated Planning and Acting*. Cambridge University Press.**

Referencia fundamental que cubre los principios teóricos y prácticos de la planificación y actuación automatizada.

Enlace:

<https://www.cambridge.org/core/books/automated-planning-and-acting/E6DE5715A2190651352DFB0869916BC3>

- **PDDL - The Planning Domain Definition Language.**

Estándar de facto para describir dominios y problemas de planificación. La documentación y versiones evolucionan;

planning.wiki es un buen recurso comunitario.

Enlace:

https://en.wikipedia.org/wiki/Planning_Domain_Definition_Language

- **Helmert, M. (2006). The Fast Downward Planning System.**

Documentación y sitio oficial del sistema de planificación Fast Downward, una herramienta clave utilizada en este trabajo.

Enlace:

<https://www.fast-downward.org/>

- **DELFI Planning System (Documentación Específica).**

Sistema utilizado para la selección del planificador basado en aprendizaje automático. La documentación detallada puede ser específica de publicaciones o del grupo de investigación. Se recomienda buscar publicaciones de los autores relevantes (e.g., Sievers, Katz, et al.) si se requiere información profunda.

Enlace:

<https://planning.wiki/guide/whatis/pddl>

- **Graphviz - Graph Visualization Software.**

Herramienta de código abierto para la visualización de grafos, potencialmente utilizada para generar el mapa mental o visualizar estructuras.

Enlace: <https://graphviz.org/>

- **Draw.io / diagrams.net.**

Herramienta de diagramación en línea y de escritorio (también integrada en VS Code mediante extensión), útil para crear mapas mentales, diagramas de flujo y otros grafos visuales.

Enlace:

<https://www.diagrams.net/> (Sitio principal)

<https://app.diagrams.net/> (Aplicación web)

- **Mermaid.**

Herramienta basada en JavaScript para generar diagramas y visualizaciones (incluyendo diagramas de flujo) a partir de texto con sintaxis similar a Markdown.

Enlace:

<https://mermaid.js.org/>

- **Visual Studio Code Marketplace.**

Repositorio de extensiones para Visual Studio Code, donde se pueden encontrar herramientas como "Draw.io Integration", "Graphviz (dot) language support", "Mermaid Editor", etc.

Enlace:

<https://marketplace.visualstudio.com/vscode>

Índice Alfabético

A

académico	29
académicos	29
Acting	29
Actions	4
actuación	29
Además	26
algoritmos	4, 5
Ambigüedad	5
ambigüedades	5
Análisis	2, 7
apoyo	28
Aprendizajes	2, 26
archivos	6, 11, 29
ASG	25, 26
aspectos	5, 6
Ausencia	5
Automated	29
automático	5, 25, 26, 30
autónomos	21
autopistas	19
autores	30
autovía	8, 9, 15, 18, 20
avances	5

B

básicos	12
Bibliografía	3, 29

C

cambios	4
camión 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 17, 18, 19, 20, 22, 23	
camiones	14
campos	21
capaces	4
capacidades	4
capas	27
características	8
categorías	4, 7, 13
ciudad	4, 7, 8, 9, 10, 11, 13, 14, 15, 17, 22
ciudades	6, 8, 12, 14, 15, 16, 17, 18, 19, 20, 23, 25
clásica	25, 26
Code	25, 26, 30, 31
competiciones	5
Competition	5
complejidad	26
complejos	6
componentes	4, 6, 7, 11
comprensión	26, 29
comunicación	5
comunidad	5
comunitario	30
conceptos	6
conceptualización	28
Conclusiones	2, 25
concretos	5, 28
condición	5
conexión	8
conjunto	5, 28
conocimiento	26
constitutivos	7
creación	5, 11, 25, 26
cruciales	4

Cuaderno	2, 21
Cumplimiento	2, 25

D

decisiones	4, 6, 21
definiciones	5
Definition	4, 29, 30
DELFI	26, 30
desarrollo	4, 5, 21
Descripción	2, 4
descripciones	8, 26
destino	10
diagramación	30
diagramas	30, 31
Dificultades	2, 26
dinámico	8
dinámicos	8
documentación	29, 30
documento	6, 28, 29
DOCX	29
Domain	4, 29, 30
dominios	4, 5, 29
Downward	25, 26, 30

E

Editor	2, 21, 22, 23, 31
efectos	4, 8, 14, 15, 28
ejecución	4, 24, 25, 26
ejemplos	17
ejercicio	26, 29
elaboración	29
elección	26
elementos	7, 29
enlaces	29
entorno	4, 11, 28
errores	26
escenario	6, 7, 28
escritorio	30
especificación	12, 28
especificaciones	29
específicos	4, 6, 17
Estandarización	5
estático	8
estructuras	6, 30
etapas	25
evaluación	5
evidencia	6
existentes	4
exploración	26
exposición	29
expresiones	4
Extensiones	6

F

falsas	7
falsos	8
fases	26, 28
Fast	25, 26, 30
fijos	7
física	4
formales	6
formalización	4, 7
formas	4, 25

formatos.....	26
fuentes.....	29
fundamentales.....	7, 13

G

generales.....	4, 6, 11
gestión	4
Ghallab.....	29
Goal.....	5
gráfica.....	25
gráfico.....	29
grafos	26, 30
Graph.....	30
Graphviz.....	30, 31

H

Helmert.....	30
Herramientas	2, 21

I

ideas	29
Importancia	2, 5
incipientes	6
industriales	6
inicio	5
Init 5.....	
instancia.....	5, 6, 10, 11, 28
instante	8
Integration.....	31
Inteligencia	4, 21
interacción	7
interfaz.....	26
Internacional	5
Introducción.....	2, 4
investigación	6, 30
IPC 5.....	

J

JavaScript.....	31
jerárquico	13

K

Katz.....	30
-----------	----

L

Language.....	4, 29, 30
Lenguaje	28
líneas	26, 29
LISP.....	5
lista	25
localizaciones	7
lógica	4
logística.....	4, 7, 10, 12, 13, 14, 19, 21
logístico.....	6, 7, 11, 18

M

manipulación.....	25
mapas	30
Markdown	25, 26, 31
Marketplace	31
mención	26
mentales	30

Mermaid	25, 26, 31
métricas	24
modelo	5, 25, 26
modelos.....	21
modularidad	5
móviles.....	7
movimiento	15, 25
movimientos.....	25

N

Nau.....	29
necesarias.....	14, 15, 26
normales.....	4

O

objetivos.....	4, 6, 16, 29
objetos	4, 7, 8, 10, 11, 12, 13, 16, 17, 22, 23, 28
obtención	26
opción	24
opciones	26
operaciones.....	4, 7, 8
operadores.....	28
operativos	21
organización	29
originales	8, 29

P

pantalla	21, 22, 23, 24
paquetes.....	6, 11, 12, 14, 16, 17, 18, 19, 20, 23, 25
parámetros	4, 8, 12
pasos	6, 24, 25
PDDL . 2, 3, 4, 5, 6, 7, 8, 11, 12, 13, 14, 15, 16, 21, 22, 23, 25, 28, 29	
PDF28.....	
permisibles.....	28
Plan.....	2, 23, 24
planes	26
Planificación	2, 4, 5, 28
planificadores	4, 5
Planning.....	4, 5, 29, 30
posibles	21, 26, 29
Práctica	3, 5, 29
prácticos	29
Precisión	5
preferencias	6
Press	29
principales.....	11
principios	29
problemas	4, 5, 21, 22, 29
procesos	4, 21, 26
publicaciones.....	30

R

razonamiento	4, 5
recurso	30
recursos.....	6, 29
red 19.....	
Referencia.....	29
reglas	4, 6, 11, 12, 16
relación	8
relevantes	5, 28, 30
Repositorio	31
representación.....	25, 26
Requerimientos.....	2, 12
respectivos.....	17
respuesta.....	5
restricciones.....	4, 6
resultante	6, 26, 29
resultantes.....	14, 15
robótica	4, 21

rutas 18

S

sección 7
 secuencia..... 4, 11, 21, 25, 26
 secuenciales..... 4
 selección.....25, 30
 Separación..... 5
 Sievers 30
 siguientes 6, 7, 8, 11
 sistemas..... 5, 21, 26
 Software..... 30
 solución.....24, 26
 STRIPS6, 12
 Studio 25, 26, 31
 System..... 30

T

También 26
 técnicas..... 25
 técnicos..... 29
 Teoría 5
 teóricos5, 29
 The29, 30

Tipos 2, 4, 12
 tomadas6
 transportista.....8
 Traverso 29
 Types4

U

ubicación 7, 8, 18
 ubicaciones 14, 20
 University 29
 utilización6

V

variedad26
 verdaderos..... 5, 8, 28
 versión 6
 Vinculación..... 2, 16
 visuales 30
 visualización 18, 26, 29, 30
 visualizaciones.....31
 Visualization 30
 vocabulario28