Desarrollo de una arquitectura reactiva y deliberativa usando planificación en el entorno de juegos GVGAI

Trabajo Fin de Grado

Autor Vladislav Nikolov Vasilev **Director** Juan Fernández Olivares

17 de julio de 2020

Departamento de Ciencias de la Computación e Inteligencia Artificial
Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación
Universidad de Granada

Índice

Introducción

Motivación

- Planificación automática como potente herramienta para la resolución de problemas.
- Integrada exitosamente en aplicaciones reales pero no en videojuegos.
- Videojuegos presentan entornos dinámicos y complejos. No se puede dar una respuesta rápida.
- Desarrollo de agentes para juegos concretos cuya componente deliberativa se basa en planificación.

Objetivo

Creación de una arquitectura en el entorno de juegos GVGAI con las siguientes características:

1. Combina componente reactiva con deliberativa basada en planificación.
2. Lo suficientemente **general** para resolver cualquier juego del entorno.

Contribuciones principales

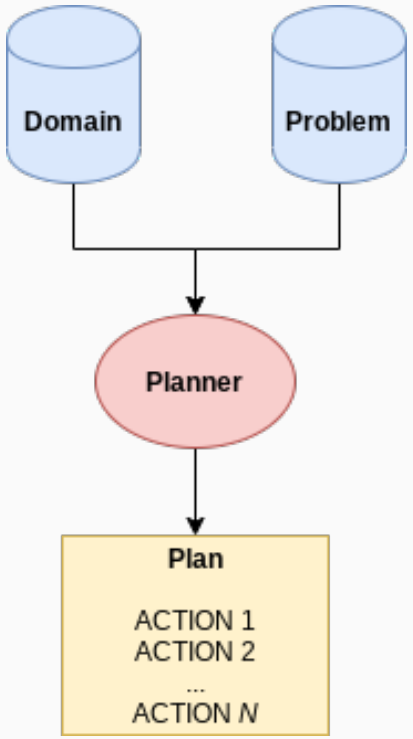
- Nuevas vías para experimentar con técnicas de planificación en GVGAI.
- Herramienta educativa.

Antecedentes

Trabajos relacionados

Propuesta novedosa, aunque se han desarrollado arquitecturas para otros juegos en específico.

Planificación



PDDL

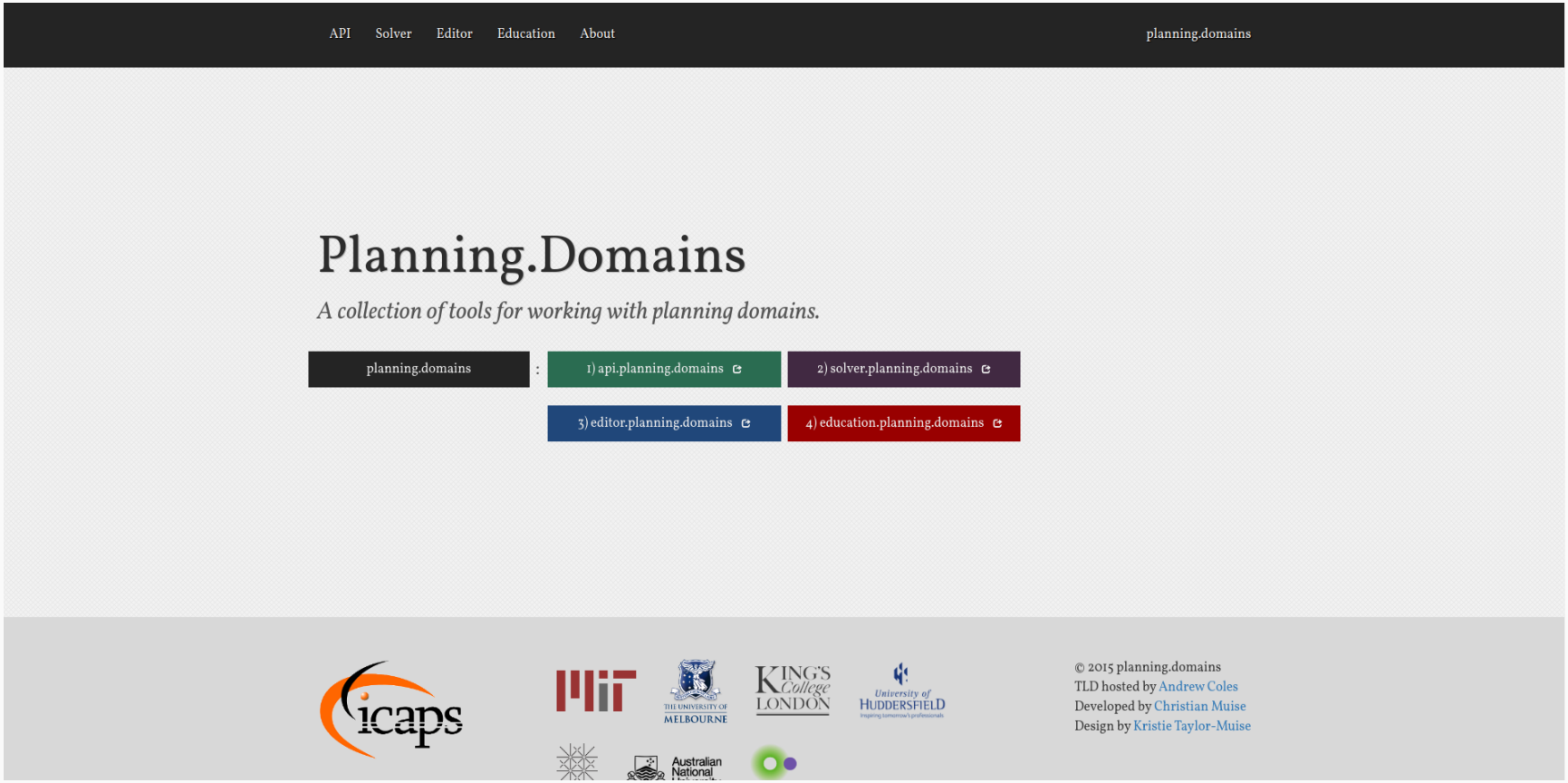
```
(define (domain vehicle)
 (:requirements :strips :typing)
 (:types vehicle location fuel-level)
 (:predicates (at ?v - vehicle ?p - location)
              (fuel ?v - vehicle ?f - fuel-level)
              (accessible ?v - vehicle ?p1 ?p2 - location)
              (next ?f1 ?f2 - fuel-level))

 (:action drive
   :parameters (?v - vehicle ?from ?to - location
                  ?fbefore ?fafter - fuel-level)
   :precondition (and (at ?v ?from)
                      (accessible ?v ?from ?to)
                      (fuel ?v ?fbefore)
                      (next ?fbefore ?fafter))
   :effect (and (not (at ?v ?from))
                (at ?v ?to)
                (not (fuel ?v ?fbefore))
                (fuel ?v ?fafter))
 )
)
```

(a) Dominio de planificación PDDL.

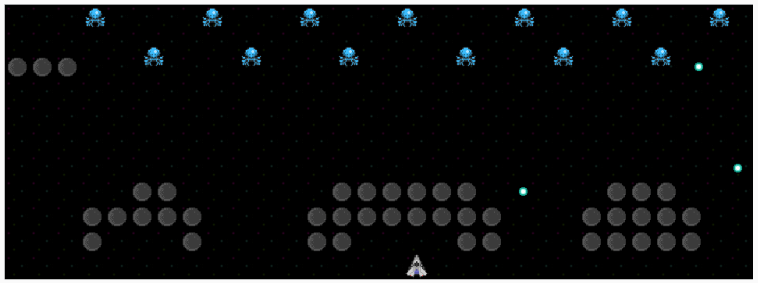
```
(define (problem vehicle-example)
 (:domain vehicle)
 (:objects
      truck car - vehicle
      full half empty - fuel-level
      Paris Berlin Rome Madrid - location)
 (:init
      (at truck Rome)
      (at car Paris)
      (fuel truck half)
      (fuel car full)
      (next full half)
      (next half empty)
      (accessible car Paris Berlin)
      (accessible car Berlin Rome)
      (accessible car Rome Madrid)
      (acessible truck Rome Paris)
      (accessible truck Rome Berlin)
      (accessible truck Berlin Paris)
 )
 (:goal (and (at truck Paris)
             (at car Rome))
 )
)
```

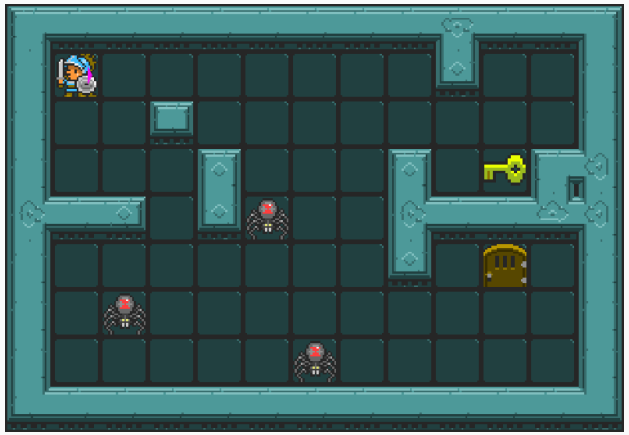
(b) Problema de planificación PDDL.

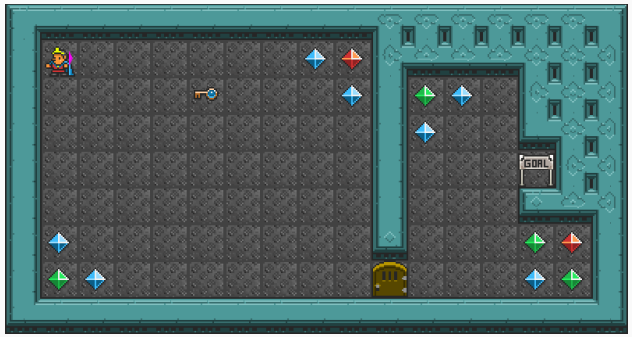
Planning.Domains



GVGAI

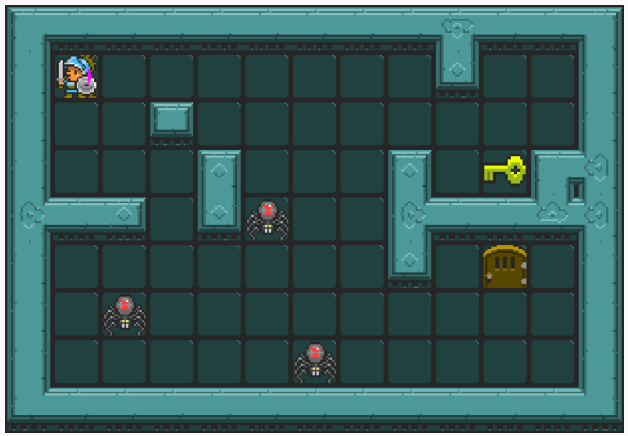






VGDL

```
wwwwwwwwwwww
wA.......w..w
w..w........w
w...w...w.+ww
www.w2..wwwww
w.......w.g.w
w.2.........w
w.....2.....w
wwwwwwwwwwww
```



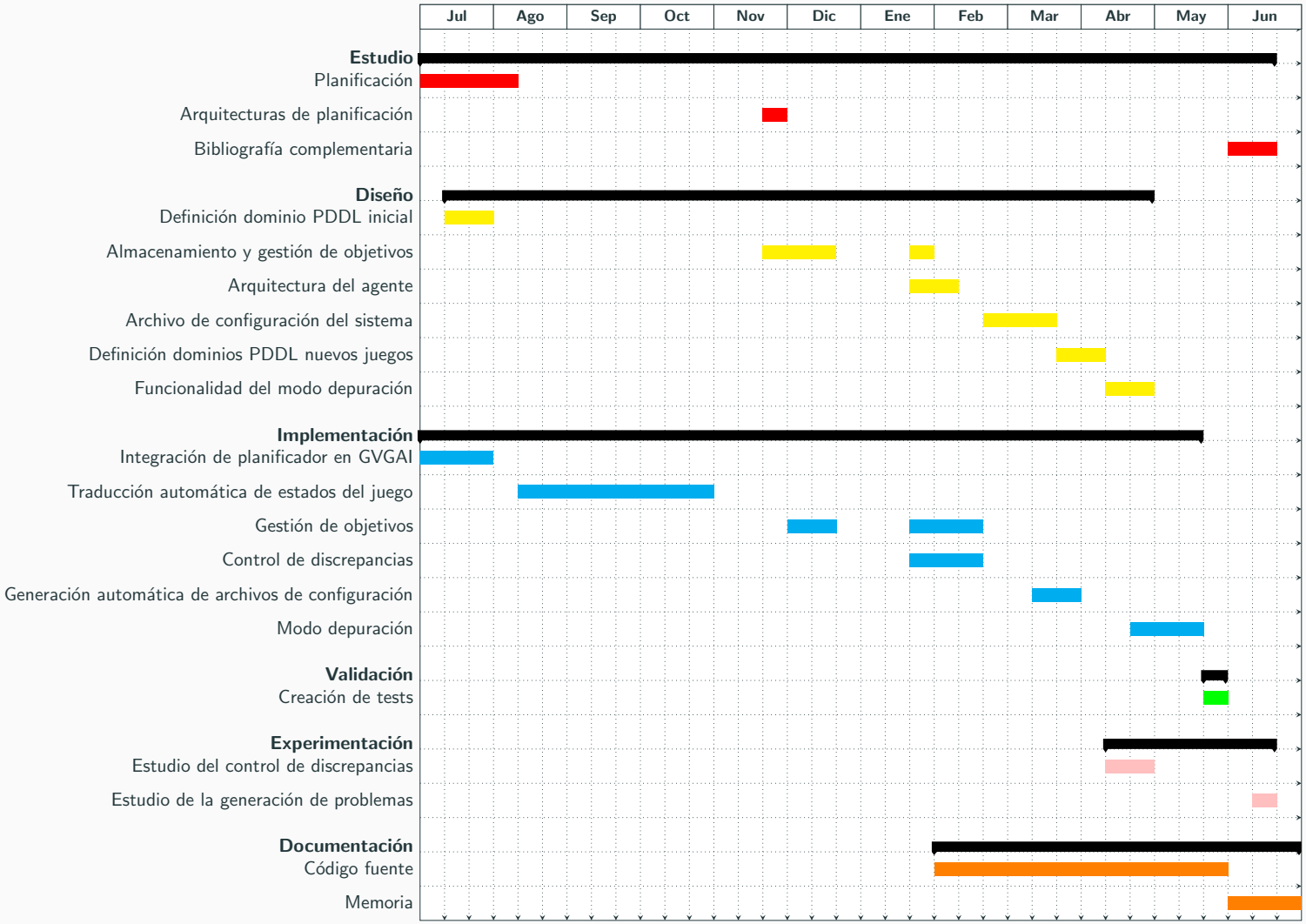
```
BasicGame
  SpriteSet
    floor > Immovable randomtiling=0.9 img=oryx/floor3 hidden=True
    goal  > Door color=GREEN img=oryx/doorclosed1
    key   > Immovable color=ORANGE img=oryx/key2
    sword > OrientedFlicker limit=5 singleton=True img=oryx/slash1
    movable >
      avatar  > ShootAvatar   stype=sword frameRate=8
        nokey   > img=oryx/swordman1
        withkey > color=ORANGE img=oryx/swordmankey1
      enemy >
        monsterQuick > RandomNPC cooldown=2 cons=6 img=oryx/bat1
        monsterNormal > RandomNPC cooldown=4 cons=8 img=oryx/spider2
        monsterSlow > RandomNPC cooldown=8 cons=12 img=oryx/scorpion1
      wall > Immovable autotiling=true img=oryx/wall3

  LevelMapping
    g > floor goal
    + > floor key
    A > floor nokey
    1 > floor monsterQuick
    2 > floor monsterNormal
    3 > floor monsterSlow
    w > wall
    . > floor


  InteractionSet
    movable wall  > stepBack
    nokey goal    > stepBack
    goal withkey  > killSprite scoreChange=1
    enemy sword > killSprite scoreChange=2
    enemy enemy > stepBack
    avatar enemy > killSprite scoreChange=-1
    nokey key     > transformTo stype=withkey scoreChange=1 killSecond=True

  TerminationSet
    SpriteCounter stype=goal   win=True
    SpriteCounter stype=avatar win=False
```

Plan de trabajo

Metodología

- Metodología de trabajo basada en **Scrum**.
- Reuniones cada 1-2 semanas:
  - Revisión de objetivos alcanzados
  - Resolución de problemas surgidos
  - Propuesta de nuevos objetivos

Temporización



Arquitectura general del sistema

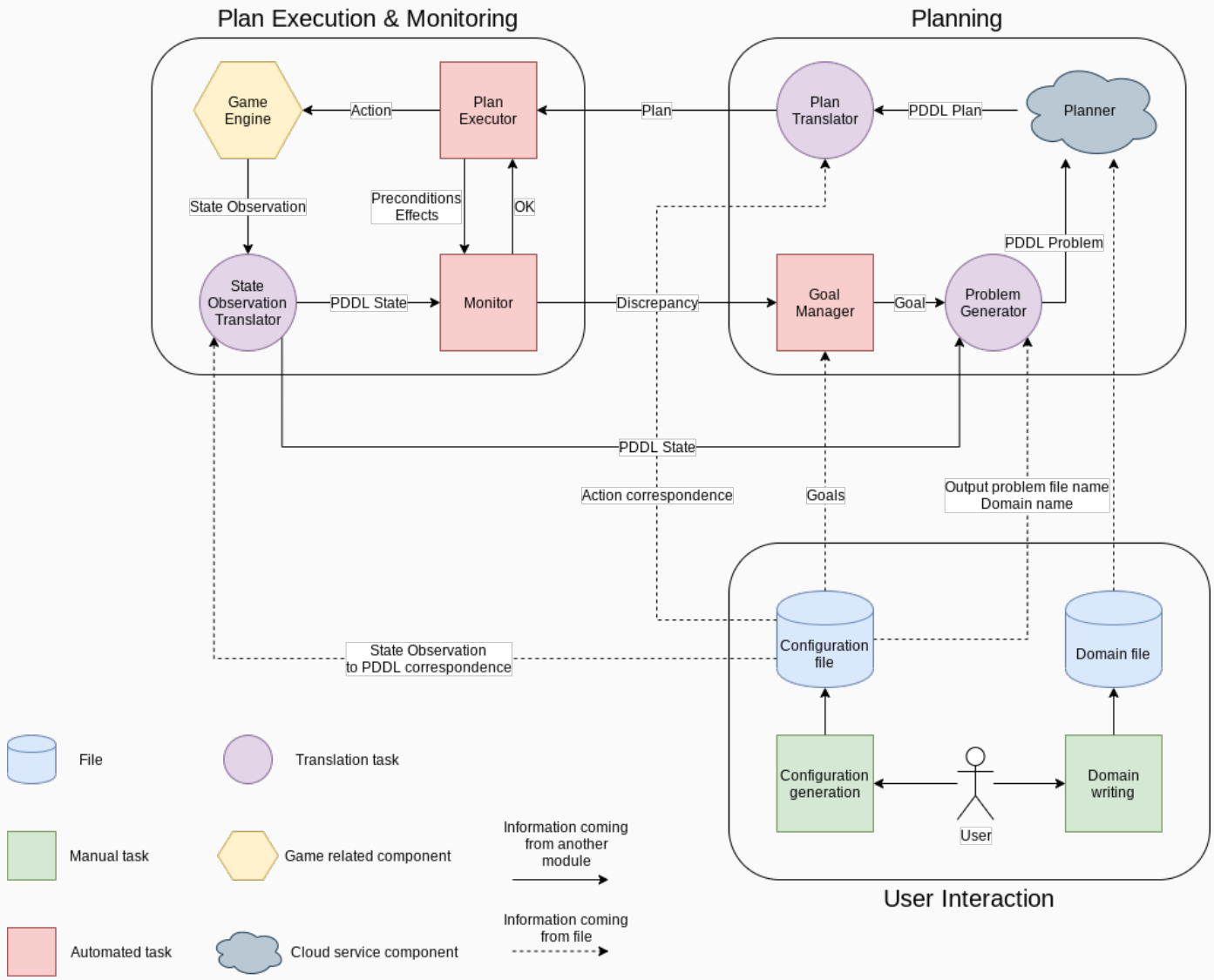
Objetivo del sistema

Dados:

1. Dominio de planificación PDDL
2. Archivo de configuración

El sistema debe resolver un nivel de un juego dado, generando para ello los problemas PDDL hasta los objetivos especificados de forma automática a partir de los estados de observación del juego.

Arquitectura general

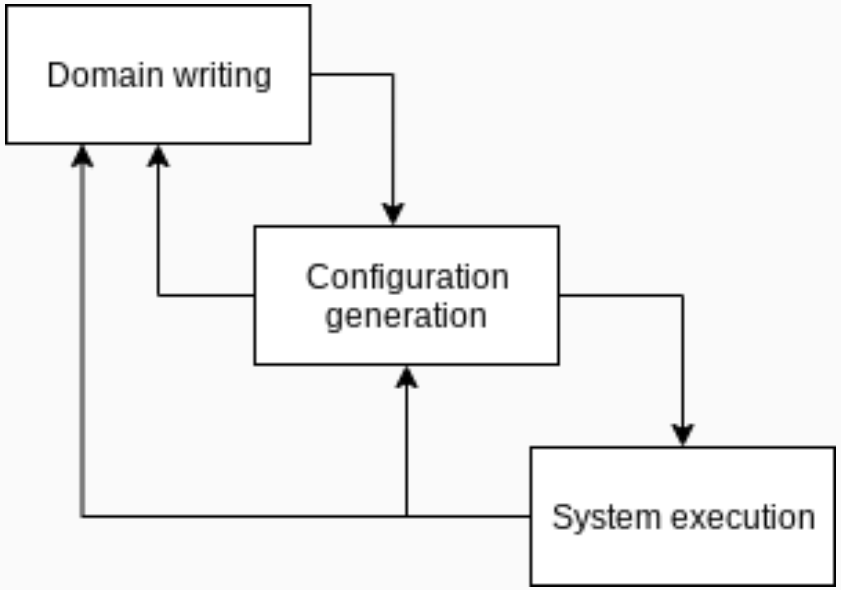


Módulo de interacción con el usuario

Descripción del módulo

- Se llevan a cabo dos tareas fundamentales:
  - Creación del dominio PDDL.
  - Creación del archivo de configuración.
- No es un componente *software*. Es más bien un proceso.
- Bajo grado de automatización.

Funcionamiento general del módulo



Creación del dominio

- Tarea manual llevada a cabo por el usuario.
- Utiliza conocimiento:
  - Jugar partidas al juego.
  - Estudiar el archivo VGDL de definición del juego.

Creación del archivo de configuración

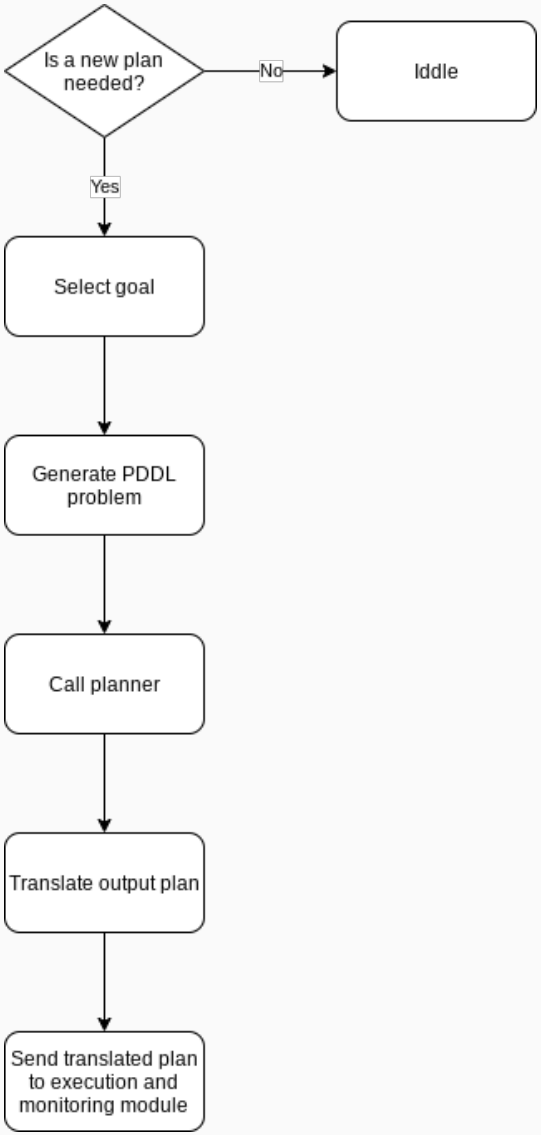
- Tarea semiautomatizada.
- Se dispone de un *script* que genera un archivo de configuración plantilla.
- Se necesitan el dominio PDDL y el archivo VGDL de definición del juego.

Módulo de planificación

Descripción del módulo

- Encargado de llevar a cabo todo el proceso de planificación y gestión de objetivos.
- Compuesto por los siguientes módulos funcionales:
  - Gestor de objetivos
  - Generador de problemas PDDL
  - Planificador
  - Traductor de planes
- Funcionamiento automatizado. Necesita dominio PDDL y archivo de configuración definidos por el usuario.

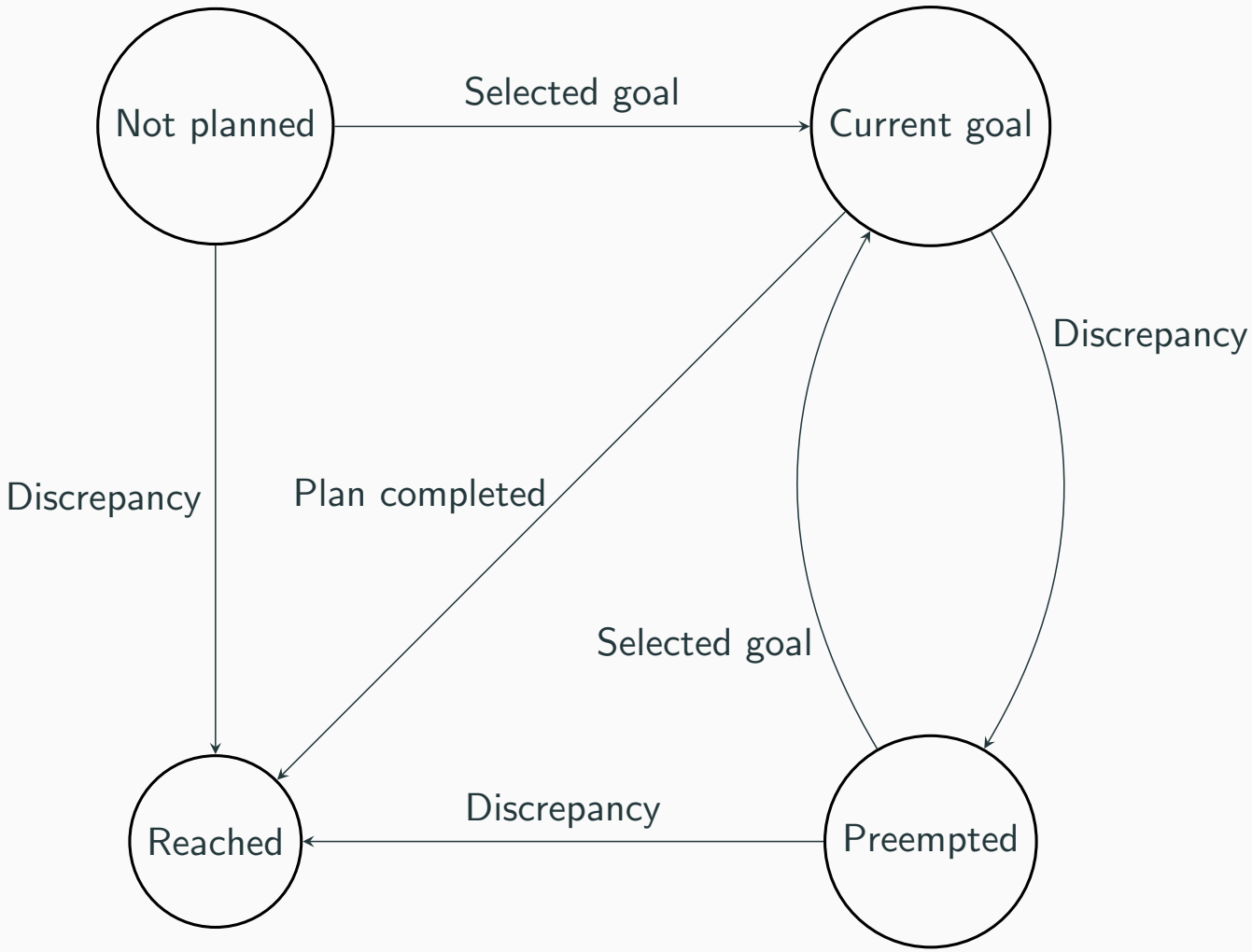
Funcionamiento general del módulo



Gestor de objetivos

- Gestionar objetivos y elegir objetivo actual.
- Estados de los objetivos:
  - No planificado
  - Alcanzado
  - Detenido
  - Actual
- Discrepancia: situación imprevista que se produce en el juego, la cual puede **detener** la ejecución del plan actual o provocar que **un objetivo se alcance prematuramente**.

Gestor de objetivos



Generador de problemas PDDL

- Genera un problema PDDL hasta el objetivo actual.
- Entradas
  - Objetivo actual
  - Estado actual del juego en formato PDDL
  - Nombre del fichero de salida
- Salida
  - Archivo de problema PDDL

Planificador

- Obtiene un plan hasta el objetivo actual.
- Entradas
  - Archivo de dominio PDDL
  - Archivo de problema PDDL
- Salida
  - Plan hasta el objetivo actual

Traductor de planes

- Traduce los planes obtenidos por el planificador a acciones interpretables por el entorno de juegos.
- Entradas
  - Plan hasta el objetivo actual obtenido por el planificador
  - Correspondencia entre acciones PDDL y acciones GVGAI
- Salida
  - Lista de acciones GVGAI hasta el objetivo actual

Módulo de ejecución y monitorización

Descripción del módulo

- Encargado de ejecutar el plan actual y de supervisar su ejecución.
- Formado por los siguientes módulos funcionales:
  - Motor del juego
  - Traductor del estado de observación
  - Monitor
  - Ejecutor del plan
- Funcionamiento automatizado. Requiere archivo de configuración definido por el usuario.

Motor del juego

- Ejecuta y visualiza el juego.
- Entrada
  - Acción a ejecutar
- Salida
  - Estado de observación actual del juego

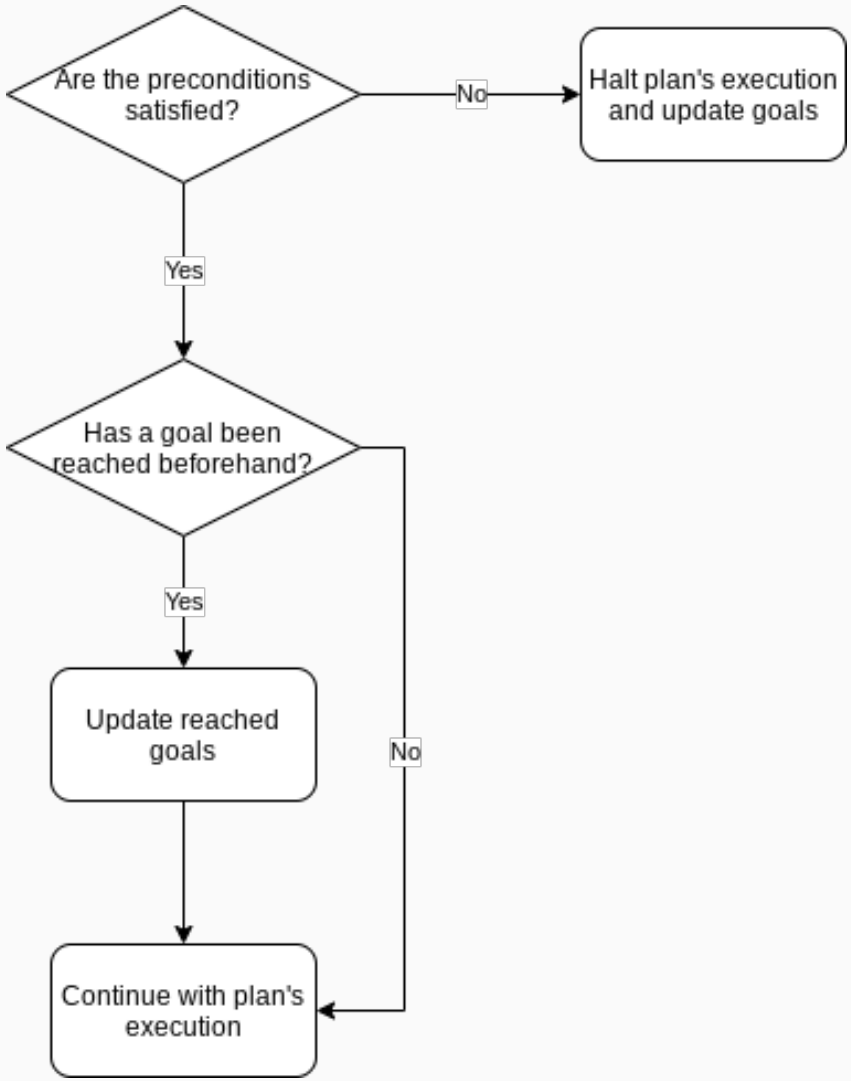
Traductor del estado de observación

- Traduce el estado de observación del juego a predicados PDDL.
- Entradas
  - Estado de observación del juego
  - Correspondencia entre los elementos del juego y predicados PDDL
- Salida
  - Estado actual del juego en formato PDDL

Monitor

- Supervisa la ejecución del plan actual.
- Estudia la existencia de discrepancias en el estado actual del juego.
- Se comunica con el **ejecutor del plan** y el **gestor de objetivos**, indicándoles la existencia de discrepancias.
- Entradas
  - Estado actual del juego en formato PDDL
  - Precondiciones de la acción a ejecutar
  - Efectos de la acción a ejecutar

Funcionamiento del monitor



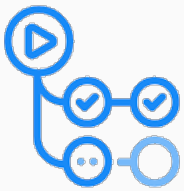
Ejecutor del plan

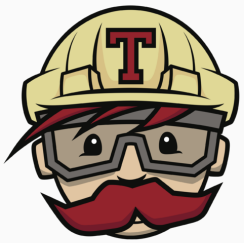
- Ejecuta el plan actual.
- Entradas
  - Plan a ejecutar
  - Señal de control enviada por el monitor indicando si el plan actual sigue siendo válido o no
- Salida
  - Siguiente acción a ejecutar del plan actual

Implementación

Tecnologías utilizadas

Maven





Experimentación

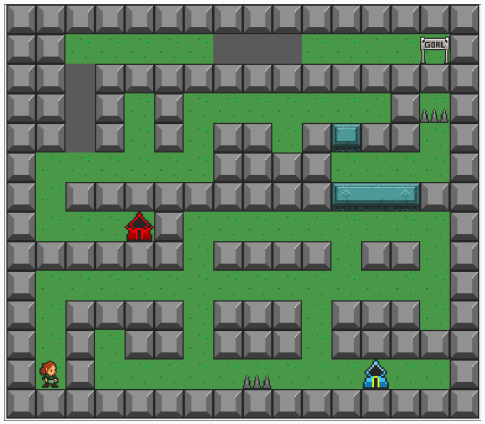
Objetivos de la experimentación

- Se quieren poner a prueba los siguientes aspectos del sistema:
  1. Capacidad de generación de problemas PDDL de forma automática.
  2. Capacidad de responder a cambios dinámicos en el entorno durante la ejecución del juego.

Juegos utilizados

(a) *Boulderdash*.

(b) *Ice and Fire*.



(c) *Labyrinth Dual*.

Generación de problemas

Juego	Nivel	Objetivos	Predicados problema inicial	Objetos problema inicial	Tiempo medio ejecución (s)	TPC[1] (s)	TTEO[2] (s)	TGAP[3] (s)	TTGP[4] (s)
Boulderdash	**0**	10	**1735**	**400**	0.4967	0.0313	0.0024	0.0038	**0.0375**
	1	10	**1721**	**393**	0.428	0.0323	0.0029	0.0039	**0.0392**
	2	10	**1717**	**391**	0.5293	0.0347	0.0034	0.0044	**0.0425**
	3	10	**1733**	**399**	0.7487	0.0327	0.0049	0.0048	**0.0423**
	4	10	**1731**	**398**	0.4403	0.0317	0.0030	0.0039	**0.0386**
Ice and Fire	**0**	3	**1215**	**391**	0.417	0.027	0.0026	0.003	**0.0326**
	1	3	**1234**	**410**	0.4827	0.0307	0.0028	0.0056	**0.0390**
	2	3	**1235**	**411**	0.522	0.0273	0.0031	0.0045	**0.0349**
	3	3	**1219**	**395**	0.4587	0.0323	0.0029	0.0061	**0.0413**
	4	3	**1210**	**386**	0.697	0.0273	0.0031	0.0055	**0.0359**
Labyrinth Dual	**0**	3	**1085**	**369**	0.4503	0.0303	0.0029	0.0043	**0.0376**
	1	2	**1069**	**380**	0.3847	0.0303	0.0027	0.0045	**0.0376**
	2	3	**1073**	**378**	0.4107	0.0287	0.0024	0.0029	**0,0339**
	3	3	**1083**	**376**	0.41	0.0297	0.0031	0.0052	**0.038**
	4	2	**1079**	**363**	0.5623	0.029	0.0029	0.0062	**0.0381**

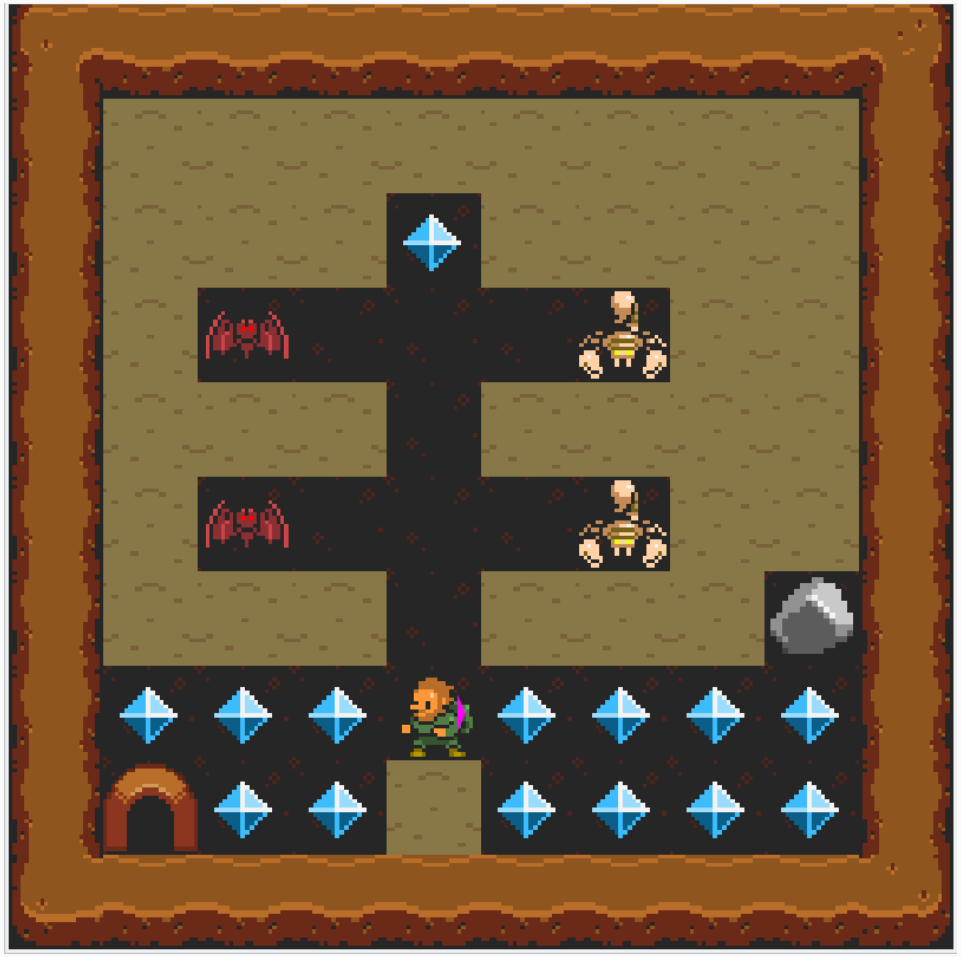
[1]Tiempo de generación de predicados de conectividad

[2]Tiempo de traducción de estados de observación

[3]Tiempo de generación de archivo de problemas

[4]Tiempo total de generación de problema

Respuesta a cambios dinámicos



Conclusiones

Conclusiones

- Resolución de múltiples juegos del entorno.
- Simplifica y acelera la generación de problemas PDDL.
- Herramienta útil para la experimentación con técnicas de planificación en GVGAI.
- Utilidad para la enseñanza: aprender sobre la planificación y cómo generar problemas de planificación.

Trabajos futuros

- Mejora del comportamiento reactivo.
- Integración de un módulo de *goal reasoning*.

FIN

¿Preguntas?